Imperial College of Science, Technology and Medicine
Department of Electrical and Electronic Engineering

Domain Adaptation for Semantic and 3D Tasks

Adrian Lopez Rodriguez

Supervised by Prof. Krystian Mikolajczyk

Submitted in part fulfillment of the requirements for the degree of PhD in Electrical and Electronic Engineering and the Diploma of Imperial College London

October 2021

Declaration of Originality

I declare that the work detailed in this thesis is solely my own work other than where appropriately referenced.

Adrian Lopez Rodriguez

Copyright Declaration

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution Non-Commercial 4.0 International Licence (CC BY-NC).

Under this licence, you may copy and redistribute the material in any medium or format. You may also create and distribute modified versions of the work. This is on the condition that: you credit the author and do not use it, or any derivative works, for a commercial purpose.

When reusing or sharing this work, ensure you make the licence terms clear to others by naming the licence and linking to the licence text. Where a work has been adapted, you should indicate that the work has been changed and describe those changes.

Please seek permission from the copyright holder for uses of this work that are not included in this licence or permitted under UK Copyright Law.

Abstract

Data is the main constraint when training Deep Learning models. Real-domain data is costly to annotate and, despite the abundance of already annotated data, training and testing on different distributions leads to low performance due to the so-called domain gap. This domain gap can be bridged with Domain Adaptation methods, which have been mostly researched within an image classification setting. Other tasks, which can be more difficult to annotate, have been less researched. We thus aim to find the efficacy of standard Domain Adaptation techniques for a set of semantic and 3D tasks.

We first investigate object detection in a multi-style target dataset setting. For this task we propose three modules based on common Domain Adaptation techniques that are tailored to the challenges of object detection (*e.g.*, negative-examples sampling or multiple local predictions per image) and the characteristics of a multi-style dataset. We hence propose to jointly use feature consistency and data adaptation with a multi-style focus to reduce the effect of the image style on the feature maps. We then use a negative sampling-aware pseudo-labelling approach to further adapt the classifiers.

Motivated by the difficulty of obtaining accurate depth ground-truth, we then investigate two depth-related tasks, *i.e.*, depth estimation and depth completion. Specifically, in our depth estimation work, we combine the natural relationship between semantics and depth to generate size-depth pseudo-labels. We also force output consistency with the depth predicted from a semantic map, which presents a lower domain gap. In our depth completion work, we analyze the noise present in a real RGB+LiDAR set-up to propose a hypothesis for the main source of artifacts present. This hypothesis is used to simulate real-like artifacts in synthetic data using multi-view projections, and also to filter the real-domain input LiDAR, which can then be used as a pseudo-label. The improved results in both tasks show that combining geometric and Domain Adaptation ideas can lead to a better adaptation.

Lastly, we research camera pose regression. For that task, we focus on generating synthetic data that can be successfully combined with real data to improve the performance of a relative pose regression model. To do so, we propose to use geometric knowledge to generate synthetic views that alleviate current dataset biases, as these biases are limiting the performance of pose regression methods.

Through the proposed modifications of common Domain Adaptation techniques (*e.g.*, data adaptation, consistency terms or pseudo-labelling) we reach state-of-the-art results for all of the proposed tasks in standard benchmarks, thus showing that these approaches are useful for a diverse set of tasks. Lastly, we discuss the trends observed in this thesis and we propose future directions that can help reduce the annotation and deployment cost of current machine learning pipelines.

Acknowledgements

First, I wish to express my gratitude to my supervisor, Prof. Krystian Mikolajczyk, for all the guidance given during these years. His ability to formalize ideas and build compelling stories has been an invaluable help. Also, to Ben and Vassilis, who provided both insightful advice and research ideas throughout our discussions. And to Tony, with whom I shared lots of memes-fueled laughter while learning about visual localization, despite R1 not enjoying our paper.

I also wish to thank my friend and labmate, Axel. We shared almost every part of the PhD (including an internship in the same research group), and spent uncountable hours building up the Deep Learning tutorials while balancing course and conference deadlines. In addition, I want to acknowledge my partners in research competitions: Edgar, Joan and Nico. We did not win any of the competitions, but at least we had tons of fun trying.

My MatchLab labmates have also been an essential part of this PhD. Our online group meetings gave me some human interaction (and a reason to wake up before 10 am) at the height of the pandemic. COVID made it hard to meet physically, but it was a blast every time we did.

On a personal note, this thesis was also possible due to the much-needed laughter shared with my friends, which was deeply needed to decompress from stressful times. I hope to celebrate with all of you at some point.

A very special word of thanks goes to Mireia. She has provided unwavering support during all these deadlines and moments of uncertainty widespread in a PhD. But more importantly, she also has been a steadfast partner in my newly-found tennis passion, which has carried me through the last months of this journey.

Finally, I wish to express my deepest gratitude to my mother. She has always been supportive throughout my whole life. I know she will enjoy seeing me dressed in a fancy purple gown.

Contents

Abstract **7**

Abstract **7**

Acknowledgements **9**

1 Introduction **33**

1.1 Contributions and Outline 38

1.2 Publications 42

2 Background **44**

2.1 Domain Adaptation 44

2.1.1 Current Domain Adaptation Techniques 46

2.2 Object Detection 51

2.3 Monocular Depth Estimation 53

2.4 Depth Completion 55

2.5 Camera Pose Estimation 58

2.6 Datasets 60

2.6.1 Object Detection 60

2.6.2 Depth Estimation and Depth Completion 63

2.6.3 Visual Localization 64

2.6.4 Summary 66

3 Domain Adaptation for Object Detection via Style Consistency **67**

3.1 Introduction 67

3.2 Method 69

3.2.1 Single Shot Detector (SSD) Supervision 70

3.2.2 Style Transfer 70

3.2.3 Feature Consistency with Style Transfer 71

3.2.4 Robust Pseudo-Labelling 72

3.2.5 Combined Domain Adaptation Loss 74

3.3 Experiments 75

3.3.1 Domain Adaptation for Object Detection Results 77

3.3.2 Limitations 81

3.4 Conclusion 82

4 DESC: Domain Adaptation for Depth Estimation via Semantic Consistency **84**

4.1 Introduction 84

4.2 Related Work 86

4.2.1 Depth and Semantic Information 86

4.3 Method . . . 87

4.3.1 Pseudo-Labelling using Instance Height . . . 88

4.3.2 Consistency of Predictions using Semantic Information . . . 90

4.3.3 Training Loss . . . 91

4.4 Experiments . . . 92

4.4.1 Main Results on Virtual Kitti→Kitti . . . 94

4.4.2 Improved Image Transfer Strategies . . . 100

4.4.3 Evaluation in Additional Settings . . . 103

4.4.4 Limitations . . . 106

4.5 Conclusion . . . 106

5 Project to Adapt: Domain Adaptation for Depth Completion from Noisy and Sparse Sensor Data 108

5.1 Introduction . . . 108

5.2 Related Work . . . 111

5.3 Method . . . 112

5.3.1 Data Generation via Projections . . . 112

5.3.2 RGB Adaptation . . . 115

5.3.3 Filtering Projection Artifacts for Supervision . . . 115

5.3.4 Summary of Losses . . . 116

5.4 Experiments . . . 117

5.4.1 Ablation Study . . . 119

5.4.2 Method Evaluation . . . 123
5.4.3 Limitations . . . 130
5.5 Conclusions . . . 131

6 Improving Camera Pose Regression via Synthetic Views 132

6.1 Introduction . . . 132
6.2 Related Work . . . 135
6.2.1 View Synthesis . . . 135
6.2.2 Transformers for Visual Localization . . . 136
6.3 Method . . . 136
6.3.1 Preliminaries . . . 136
6.3.2 View Selection and Synthesis . . . 139
6.3.3 Localization Pipeline . . . 142
6.3.4 Loss . . . 142
6.3.5 Implementation Details . . . 143
6.4 Results . . . 147
6.4.1 Comparison with State-of-the-Art . . . 148
6.4.2 Ablation Studies . . . 149
6.4.3 Inference Speed . . . 156
6.4.4 Limitations . . . 158
6.5 Conclusion . . . 158

7 Conclusions and Future Work **160**

7.1 Limitations 162

7.2 Future Directions 163

7.3 Epilogue 165

Bibliography **165**

List of Tables

2.1 Number of available object instances in the dataset for each of the classes. VOC refers to the training and validation split from PASCAL VOC07 and VOC12. The rest refer to both train and test, which follow a 50/50 split. 61

3.1 Object detection results in Average Precision (%) in the domain Clipart1k. We report results for different combinations of the loss proposed in Section 3.2, and we use $\mathcal{S}$ for source domain data, $\mathcal{T}$ for target, *DTI* for domain transferred images, *ST* for pixel level adaptation using style transfer, *C* for consistency loss, *PL* for the pseudo-labelling method from [140] and *RPL* for robust pseudo-labelling († means no style transfer applied to $\mathcal{T}$ examples). Methods marked with * use a Faster R-CNN with images of 600 pixels on the shorter side. Our approach brings significant improvements over baseline and other state-of-the-art methods from [140, 330, 51, 286]. . . . 75

3.2 Object detection results in Average Precision (%) in Watercolor2k and Comic2k. . . 76

3.3 Object detection results in Average Precision (%) in driving settings for two different settings, Sim10k→ Cityscapes and Cityscapes→ Foggy Cityscapes. 80

4.1 Results for Virtual KITTI→KITTI in KITTI [97] Eigen [82] split. For a fair comparison, we use the official pretrained models given by *T²Net* and *S2R-DepthNet* to recompute the results using median scaling. *S³Net* is the best reported result in [54] not using any semantic maps at test time, whereas *S³Net (Test Semantic)* uses semantic maps both at train and test time. S2R-DepthNet [49] is a domain generalization method that does not use any KITTI data during training. 92

4.2 Ablation study of DESC for Virtual KITTI→KITTI in Eigen split [82] capped at 80m. *Img.* refers to using image translation, *Ins.* to using instance-height pseudo-labels (Section 4.3.1) and *Con.* to the consistency of predictions constraint (Section 4.3.2). . 93

4.3 Results on KITTI Eigen split (80m cap) for methods using stereo data in KITTI. Due to an evaluation error in [374], results from GASDA are recomputed using the official pretrained models. We include a state-of-the-art stereo-trained method, *Monodepth2* [104]. 95

4.4 Results on KITTI Eigen split (cap 80m) when varying either the weight of the semantic consistency loss $\lambda_{\mathcal{T}}$, and the number of training iterations for the last step of training. Default values used in DESC are $\lambda_{\mathcal{T}} = 1.0$ and 20000 training iterations. . . 99

4.5 Results of DESC on the KITTI Eigen test split with different image transfer modules. 101

4.6 Results on Make3D (70m cap) [298] using the same central image crop as [256]. All of the methods have been trained on Virtual KITTI→KITTI, except for S2R-DepthNet, which is only trained on Virtual KITTI. For non-median scaled results of SharinGAN, GASDA and T²Net, we report the results given in [256], and we compute the median scaled results using the evaluation code given in [256] along with the official pretrained models given by each method. 103

4.7 Results on the KITTI Eigen test split (cap 80m) when using a semi-supervised setting with 1000 labeled KITTI images. No median scaling performed during evaluation for this experiment. Results for *T²Net*, *Real + Syn* and *ARC* taken from [378]. 104

4.8 Results on the KITTI 2015 stereo 200 training set disparity images [223, 97]. We include *Monodepth2* [104], the state-of-the-art stereo method trained only in KITTI. Results for non-stereo trained methods (*T²Net*, *S2R-DepthNet* and *DESC*) are reported with median scaling. 105

4.9 Cityscapes→KITTI results, evaluated in KITTI [97] Eigen split (80m cap). *Struct2Depth (M+R)* [40] uses three consecutive frames for refinement. 105

5.1 Ablation study on the KITTI 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. *Real GT Supervision* is the original result from [333] trained with a MSE loss, whereas *Real GT + Synthetic GT* refers to our result combining both CARLA and KITTI ground-truth and using a BerHu loss. 119

5.2 Results on the KITTI 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. 121

5.3 Comparison of results on the KITTI selected validation set and the official online test set. *DA Base* is our Domain Adaptation baseline formed by CycleGAN [382] + LiDAR Masks. RMSE and MAE are reported in mm, and iRMSE and iMAE in 1/km. We compare against the supervised and self-supervised S2D [211] variants and the self-supervised VOICED [343], the unsupervised, self-supervised and supervised variants *DDP* [357], and the supervised FusionNet [333]. The unsupervised *DDP* [357] refers to only using Virtual KITTI data without using any KITTI data. *DA Base* is our Domain Adaptation baseline, with adversarial adaptation with a discriminator at the output (*D. Out*) or in the features (*D. Feat*). *Ours* uses FusionNet architecture, *Ours-S2D* uses S2D architecture and *Ours+SS* uses the video self-supervision from [211]. Bolded results refer to best performance among those methods not using the semi-dense KITTI ground-truth. 122

5.4 Semi-supervised results on the KITTI 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. 125

6.1 Sanity check on *Street*. For the first and second experiments we train on test views with real images and synthetic views generated with training images respectively. The last row is the coarse prediction with image retrieval. 139

6.2 Comparison of visual localization methods on *Cambridge Landmarks* [154]. We report the median errors in translation (m) / rotation (°). The four types of methods are image retrieval (IR), structure-based (3D), absolute pose regression (APR) and relative pose regression (RPR), which our method belongs to. We highlight the best results for IR, 3D and all regression methods with red, blue and bold respectively. [(-) no results reported (✗) method fails to train.] 146

6.3 Comparison of visual localization methods on *7Scenes* [103]. We report the median errors in translation (m) / rotation (°). The four types of methods are Image Retrieval (IR), structure-based (3D), Absolute Pose Regression (APR) and Relative Pose Regression (RPR), which our method belongs to. We highlight the best results for IR, 3D and all regression methods with red, blue and bold respectively. 147

6.4 Ablation study with respect to different components of our localization pipeline. We compare training with real image pairs only and our *In-Distribution* approach. For each case, we also compare the use of MLP vs. Transformer. 149

6.5 Results when using limited amount of data jointly with our synthetic images. We use *Out-of-Distribution* to generate a single novel view per training image for different amounts of data, and we use the resulting pairs for training. 150

6.6 Median distance between the test query images and the closest synthetic pose generated during training. We generate a synthetic pose following either our *In-Distribution* or our *Out-of-Distribution* approach for 1200 epochs following our experimental settings. To retrieve the closest synthetic pose, we use as metric a weighted sum of translation and rotation distance (*i.e.*, $\lambda t_d + r_d$), where the weight λ is obtained using the ratio of rotation and translation errors in Table 6.2 for each scene and method. For example, for the reported closest synthetic pose for *Out-of-Distribution King's College* the weight λ is set to $\lambda = \frac{0.51}{0.35}$. 152

6.7 Additional quantitative results in *7Scenes* [103]. *Real* refers to training with only real data, and we include results for our *Out-of-Distribution* method (*O.o.D*) when using during inference either the GeM top-1 NN or a randomly chosen database image as our NN. Reported results are the median translation (m) / rotation (°) errors. 152

6.8 Results in *Shop Façade* and *Shop Façade Synthetic*, along with the *Relative Performance* difference between evaluating on *Shop Façade* vs. *Shop Façade Synthetic*. *Baseline Synth.* is our method trained with only with synthetic images from our *Out-of-Distribution* module without multi-image accumulation (*i.e.*, only projecting a single database image) and only using COLMAP depth. + *Multi-image accumulation* adds to *Baseline Synth.* the accumulation of 10 database images to generate the synthetic images. + *Fused Depth* further adds to *Baseline Synth.* + *Multi-image accumulation* a fusion of the predicted depth with the COLMAP depth to generate denser depth maps. + *Real (Full)* is our full *Out-of-Distribution* pipeline, which combines 25% of fully real images to the training batch with the synthetic images generated with multi-image accumulation and depth fusion. 153

List of Figures

1.1 Examples of data annotation. Crowdsourcing semantic labels is more time consuming for denser tasks compared to image classification. Depth labels can be coarsely crowdsourced (*e.g.*, binary annotation of *closer* and *further* [48]) or generated using image collections from a single scene [192]. However, for an accurate metric-scale depth map we need on-site collection with special devices, where the election of the depth device presents trade-offs (*e.g.*, cost, density or accuracy). 35

1.2 Synthetic data advantages. Synthetic data can produce a vast amount of detailed multimodal data (left image [131]), can safely simulate long-tail events hard to encounter in real life (center image [313]), and is privacy-friendly (right image, faces are synthetic [369]). 37

1.3 Real (KITTI [97]) and Synthetic (Virtual KITTI [92]) examples showing a visual domain gap. The real and synthetic images represent the same semantic scene, however the differences in illumination and textures, among others, hamper the generalization on the real data of a model trained with synthetic data. 37

2.1 Overview of Domain Adaptation techniques. The yellow blocks refer to techniques that are usually applied in a Domain Adaptation work, which we describe in this chapter. Dashed lines show possible techniques applied, whereas solid arrows refer to the standard modules used in Unsupervised Domain Adaptation works. RGB and segmentation examples are taken from [53]. 46

2.2 Object detection problem. Given an image, we aim to retrieve a bounding box and class for all instances of objects (usually from a predetermined set of classes) in an image. Deep Learning object detectors can be classified into two-stages or one-stage detectors depending on whether they use a refinement step with feature pooling. . . . 51

2.3 Monocular depth estimation problem. For a given individual RGB image, we aim to estimate a dense depth map, which can be in a known (*e.g.*, metric) or unknown scale depending on the setting. Supervision for this task can be in the form of a given ground-truth, or leveraging video or stereo sequences during training (which we assume are not available at inference time). 53

2.4 Depth completion problem. Given a sparse depth, we aim to output a fully dense depth map. The depth sparsity pattern affects the approaches used, as Structured Light or Time-of-Flight devices (top left) contain a mostly dense depth with large missing areas, whereas LiDAR patterns (bottom left) resemble irregular grids. 56

2.5 Camera pose estimation problem. We aim to estimate the camera pose where a single query image was taken from. We focus on regression-based methods, which can follow either an Absolute Pose Regression (APR) or a Relative Pose Regression (RPR) approach. In APR, the 6 Degrees-of-Freedom camera pose is directly estimated from a single image using a scene-specific coordinate frame (solid lines). RPR methods first retrieve a nearest neighbor with a known pose from the image database, which is combined with the query image to estimate a relative pose (dashed lines). 58

2.6 Top: Sample annotated images from the object detection Cross-domain Dataset from [140]. Bottom: Domain Transferred Images from [140]. 61

2.7 Samples from Clipart1k, Comic2k and Watercolor2k showing the diversity of styles inside a specific domain. 62

2.8 Examples of images from the driving datasets used in this thesis. Although they represent a similar scene, the datasets contain differences in illumination, weather and contrast among others. 62

2.9 Examples of images from camera pose estimation datasets. 65

3.1 Overview of the unsupervised Domain Adaptation object detection task. We have a labeled source set $\mathcal{S}$ and an unlabeled target set $\mathcal{T}$ that shares classes with $\mathcal{S}$. We aim to use both sets to train a detector with good accuracy in $\mathcal{T}$. 68

3.2 Overview of the proposed approach for Domain Adaptation of object detectors. We first make the detector robust to low-level variations by using a style transfer method for pixel-level adaptation of the source images, and then we force a high-level feature consistency between the style transferred image and the original image. As a final step, we pseudo-label $\mathcal{T}$, and finetune the model following the same pixel-adaptation and feature consistency method with both images from $\mathcal{S}$ and $\mathcal{T}$. 69

3.3 Left: Instances and unique classes per image in PASCAL VOC07/VOC12. Right: Precision and Recall curves for different recognition thresholds for $h(b)$ for the test set of PASCAL VOC07 and the training set of the *DTI* images with a model trained in PASCAL VOC07/VOC12. 73

3.4 Qualitative results. Left: Examples of style transferred images and corresponding CycleGAN generated image in [140]. Right: Examples of object detections in the target domain. 78

3.5 Left: Effect of the threshold used for pseudo-labelling in Clipart1k. Right: Effect of varying λ_i values in Clipart1k. 79

3.6 Examples of images from driving datasets, and style transfer in those datasets. Left image is the content image, center image is the style image, and right image is the style-transferred image. 81

3.7 Examples of failed detections of our final model in the target domain for the adaptation from PASCAL VOC to multi-style artistic datasets. 82

4.1 Overview of the data available and proposed supervision. The source domain $\mathcal{S}$ contains both RGB and ground truth depth data, and the target domain $\mathcal{T}$ contains RGB data only. We train a depth estimation model to achieve high performance in $\mathcal{T}$ by leveraging semantic annotations to introduce semantic consistency in $\mathcal{T}$. The semantic annotations are obtained using a panoptic segmentation model trained with external data. 85

4.2 Overview of the approach. We train a depth estimation network G_D with both target $\mathcal{T}$ and source $\mathcal{S}$ images. Source images are adapted to the style of the target images. For $\mathcal{S}$, we use ground truth supervision, while we enforce consistency with semantic information in $\mathcal{T}$. The consistency is enforced with (1) predictions from a second network G_S trained with edges and semantic maps as input, and (2) depth pseudo-labels formed using an instance height $\hat{h}$ predicted by G_h. Both G_S and G_h are trained using ground truth data from $\mathcal{S}$. The architecture of G_h is given in the top right. We use ReLU between the layers of G_h. 88

4.3 Qualitative results in KITTI for models trained on Virtual KITTI→KITTI. Ground truth depth is linearly interpolated for visualization. Green bounding boxes refer to areas of the prediction more accurate compared to the corresponding red bounding boxes. 94

4.4 Regions with large error for our model trained on Virtual KITTI→KITTI. 94

4.5 Qualitative results when using G_S, which takes as input the concatenation of a semantic segmentation (second column) and an edge map (third column) to output a depth map (last column). 97

4.6 Qualitative results in KITTI for models trained on Virtual KITTI→KITTI with stereo supervision in KITTI. Bottom row corresponds to a center crop of the original image. 98

4.7 Absolute relative error vs. ground truth distance for three different methods in the KITTI Eigen test split. We also include a histogram of the ground truth distribution. Left axis shows both absolute relative error (line plot) and ratio of ground truth values (histogram). 100

4.8 Distribution of absolute relative error averaged over all pixels for different classes over the KITTI Eigen test split (cap 80m) for three different methods trained on Virtual KITTI→KITTI. The percentage of ground truth depth corresponding to each class is given below each class. 100

4.9 Qualitative results for three different image transfer strategies. Virtual KITTI images are transferred to either a style matching KITTI images using a T^{2}Net-based approach or to an intermediate shared domain with the transferred KITTI images using either a SharinGAN [256] or S2R-DepthNet [49] approach. The original single-channel S2R-DepthNet images are mapped to RGB using a colormap and logarithmic mapping. 101

4.10 Transfer of KITTI images to the intermediate shared domain employed by SharinGAN [256] and S2R-DepthNet [49]. The original single-channel S2R-DepthNet images are mapped to RGB using a colormap and logarithmic mapping. 102

4.11 Qualitative results in Make3D for T^{2}Net and DESC. 103

5.1 Left: 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. Right: Overview of the multicamera set-up in CARLA used to simulate the real projection LiDAR artifacts. 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. 110

5.2 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. The method has two training 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 $\mathcal{R}_p$ and red points are dropped. The *Image Translation* network is pretrained using a CycleGAN approach [382]. 111

5.3 Distribution of errors in the input depth for the KITTI training set and for our collected CARLA dataset with our projection module. 112

5.4 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. 113

5.5 Qualitative results with different training methodologies. *Bernoulli* refers to training using the Bernoulli sparsified depth x_D^{sB}, *LiDAR Mask* to training using the depth sparsified with real LiDAR masks x_D^{sM} and *Ours* to our full pipeline that uses both LiDAR masks and projections. Both rows show projection artifacts, which *Ours* deals with correctly. 120

5.6 Qualitative results when using two different image transfer strategies. We apply to the original CARLA images (first row) either a CycleGAN approach (second row) or a FDM approach (third row). 121

5.7 Qualitative results on the official KITTI test set for *SS-S2D*, *VOICED* and our method with video supervision *Ours+SS*. Red rectangles show zoomed-in areas. 123

5.8 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. training iterations in the 2nd step, evaluating every 400 iterations, and using a moving average of 25 values with further averaging of 3 runs. 125

5.9 RMSE in the second step of training when varying the parameters of the filter used to obtain the reliable points $\mathcal{R}_p$. In the *left image* we vary the object thickness θ with window size set to $w_p = 16$ pixels. In the *right image* we vary the window size w_p, with object thickness fixed to $\theta = 0.5$ m. *1st Step* is the performance obtained after the first step of training, *i.e.*, only using synthetic data. *No F.* refers to not using any filter in the real sparse depth used for supervision during the second step of training. . 126

5.10 Qualitative results on PandaSet [299] for our *DA Baseline* and full method (*Ours - Full*) trained in CARLA and KITTI. RGB images also show overlaid the sparse depth input. 127

5.11 Left: Distribution of depth ground-truth values in the KITTI validation set and the CARLA training set. Right: MAE vs. ground-truth depth values in the KITTI validation set and the CARLA training set for our final model. 127

5.12 Error and data distribution depending on the detected (with Mask R-CNN [121] pre-trained on COCO [345]) class on the KITTI selected validation set for the crop of 1216×256. *MAE* is the mean absolute error, *Noise Input* is the mean absolute error in the input depth values, *Ratio Depth* is the ratio of the depth GT values available divided by the total amount of ground-truth values, *Ratio RGB* is the ratio of pixels representing a class divided by the total amount of pixels. 128

5.13 KITTI ground-truth overlaid on the RGB image. The red rectangles show challenging areas with missing ground-truth due to the ground-truth acquisition process used in [331]. 129

5.14 Left: RMSE obtained when varying the sparsity level by sampling the input sparse depth with probability p_S. Right: Methods supervised with KITTI ground-truth (top) expand the predictions around the edges compared to our method (bottom), which uses synthetic supervision. 129

5.15 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 a dropping of points due to low-reflectance black surfaces. 130

6.1 Challenge facing CNN methods for camera pose regression. We show a bird's-eye view of the 3D model of a street. The (1) and (2) dotted lines and arrows are the trajectories and orientations of two separate training sequences. As the training data is heavily biased towards specific views, CNN methods do not extrapolate well to the test pose in (3). Our pose regression method alleviates this by learning from (4) synthesized views sampled from an unbiased pose distribution. 133

6.2 Distribution of absolute *yaw* rotations of *Kings* vs. *Street* in the Cambridge Landmarks [154] dataset. The four peaks in *Street* show there may exist large bias in the data collection process compared to the more uniform *Kings* distribution. 138

6.3 Illustration of view selection and synthesis for training. Left: We first sample a target pose $\hat{C}$ by a Gaussian (opaque green cloud) based on an existing pose C_i. Right: We produce synthetic images from $\hat{C}$ by combining densely-predicted but scale-agnostic depths with the sparse metric MVS depths of the nearest images, and re-projecting to the target pose. 138

6.4 Our localization pipeline. The image pair - query and nearest neighbor (N.N.) - are represented by blue and red arrows. During training, the N.N. is synthesized from the process described in Section 6.3.2. In inference, N.N. is determined by an image retrieval method. The image pair is fed into our *Pose Regression Module*, which contains a Transformer that predicts the relative pose between the two. 141

6.5 Architecture of our *Pose Regression Transformer*. 145

6.6 Median error in predictions depending on the distance to the NN for two different scenes (*King's College* and *Street*) and for the models trained only with real data and the models trained with our *Out-of-Distribution* method. We include both rotation and translation errors, along with the ratio of test query-NN pairs (shown in *Ratio Images*) for the different distances shown. 151

6.7 Examples of novel views synthesized with our approach. Using only the *MVS Depth* limits the quality of the synthesized image due to the depth sparsity. Our *Fused Depth*, presented in Section 6.3.2, allows for denser synthetic views. 154

6.8 Synthetic images used during training in the Cambridge dataset. The images are randomly selected from our method, we filtered the images with less than 30% non-blank pixels. 154

6.9 Synthetic images used during training in the 7Scenes dataset. The images are randomly selected from our method, we filtered the images with less than 30% non-blank pixels. 155

6.10 Qualitative results with our *Out-of-Distribution* trained model. For visualization purposes, we synthesize an image in the pose predicted by our model (*Prediction* columns). 156

6.11 Examples of incorrect NNs retrieved by GeM in *Street*. 157

6.12 Examples of incorrect predictions in *Street*. For visualization purposes, we synthesize an image in the pose predicted by our model (*Prediction* columns). 157

Chapter 1

Introduction

> "Data! Data! Data!" he cried impatiently. "I can't make bricks without clay."
>
> *Sherlock Holmes*
>
> *Arthur Conan Doyle*

In the past few years, we have witnessed an explosion in the field of machine learning, specifically in Deep Learning methods, spearheaded at the beginning of the 2010s by Convolutional Neural Networks (CNN). Deep Learning approaches have shown astonishing results, with CNN-based methods significantly improving upon the state-of-the-art in tasks such as classification [167], object detection [102] or image segmentation [14]. The potential shown by Deep Learning has spurred the growth of machine learning conferences (*e.g.*, 9454 paper submissions in NeurIPS 2020 *vs.* 1678 in NeurIPS 2014 [349]) and has led corporations to make large investments in AI, *e.g.*, it is estimated that companies invested $50.1 billion in AI only in 2020 [65].

Reasons for the Deep Learning Rise. The popularization of Deep Learning is due to the convergence of different factors. First, Graphics Processing Units (GPUs), which had been developed for years mainly for gaming and computer animation purposes, provided fortuitously (see the Hardware Lottery discussion in [128]) almost the perfect specialized hardware for Deep Neural Networks due to their power and parallelization capabilities. CUDA, which was released in 2007, delivered an easier platform to leverage these GPUs for parallel programming. Secondly, improvements in the Deep

Learning field (*e.g.*, Adam [162]) and the development of popular packages, such as PyTorch [247] or Tensorflow, [1] made Deep Learning methods more stable and easier to use. However, what is truly at the core of this machine learning revolution? **Data**. Specifically, a big catalyst was the release in 2009 of the largest annotated dataset up to that point, ImageNet [71], containing 3.2 million labeled images. Approaches used before Deep Learning (*e.g.*, Support Vector Machines [66] or Decision Forests [69]) reached a limit on the learning capability when increasing the training data. Conversely, Deep Learning methods were able to leverage large amounts of data to improve their performance, as shown in the results of the first CNN used in the ImageNet contest in 2012, AlexNet [167], which improved significantly upon classical methods.

More Data means better Performance. In the current Deep Learning paradigm, data (paired with enough model capacity) trumps everything. For a model with sufficient capacity, increasing the amount of data usually translates into better performance [318, 31]. For instance, coupling a 175 billion parameters generative language model with 500 billion training tokens [31] achieves outstanding language generation capabilities. Computer Vision has also followed the trend of pairing large data volumes with a large model capacity to obtain better results. JFT-300M [318], a dataset of more than 300 million images, has been used to pretrain models for multiple tasks. JFT-300M pretraning has shown that in a supervised setting the performance of a CNN keeps increasing in a logarithmic trend with the number of training examples. On that note, Visual Transformers [77] leveraged the JFT-300M dataset to obtain better results than Convolutional Neural Networks in image classification, showing that the inductive biases of CNNs (weight-sharing and locality) can even be discarded when your training data is sufficiently large. All these results show that with large amounts of data we can achieve outstanding (and even surprising) results.

Annotation Limitations. The increased performance when using large datasets has led the research community to annotate large datasets for a wide variety of tasks. However, Deep Learning models show a significant brittleness when tested in distributions different to the training distributions. For example, training a CNN [167] in SVHN [234] and testing in MNIST [174], which are both datasets representing digits, yields a classification accuracy of 71.7% [89], while reaching 99.6% when training directly on MNIST. This brittleness suggests that to obtain good performance in real-

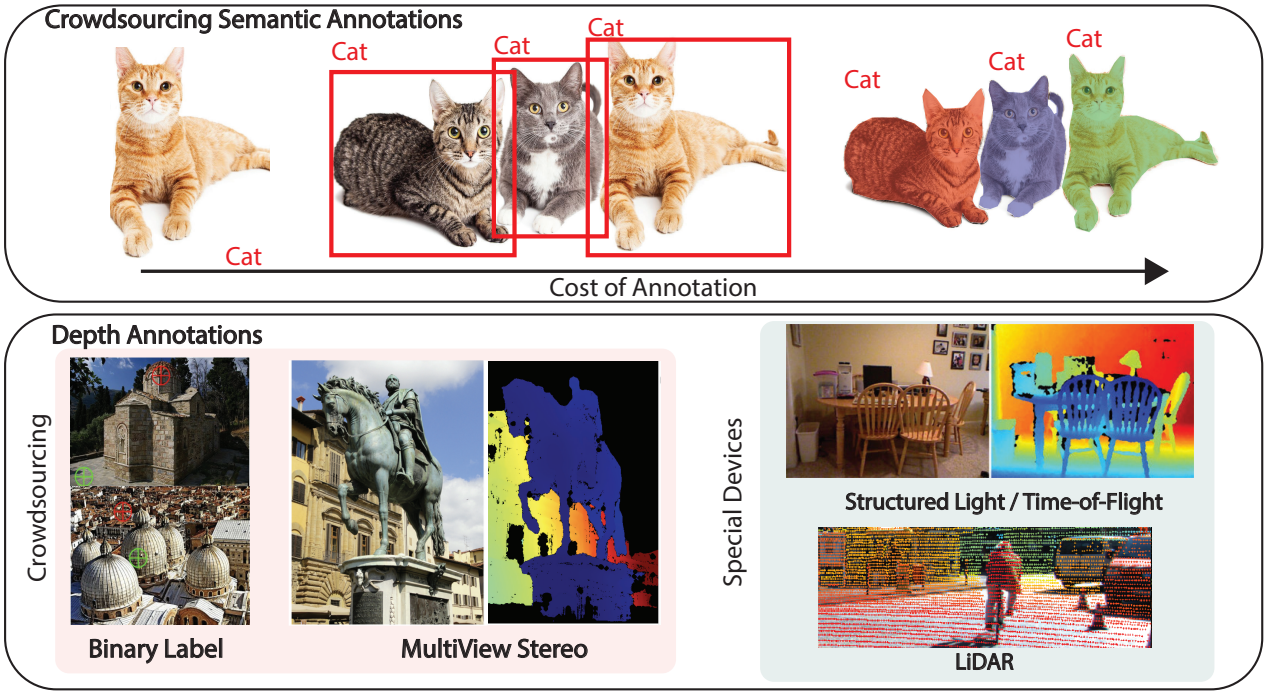


Figure 1.1: Examples of data annotation. Crowdsourcing semantic labels is more time consuming for denser tasks compared to image classification. Depth labels can be coarsely crowdsourced (*e.g.*, binary annotation of *closer* and *further* [48]) or generated using image collections from a single scene [192]. However, for an accurate metric-scale depth map we need on-site collection with special devices, where the election of the depth device presents trade-offs (*e.g.*, cost, density or accuracy).

world applications, we should keep annotating every different data distribution that our model will encounter. Nevertheless, data annotation is costly, especially for some tasks as Figure 1.1 shows. For instance, pixel-wise annotation for Microsoft COCO [195] required $\sim 80s$ per instance, or more than 1.5 hours per image in the Cityscapes driving dataset [64]. Furthermore, in some tasks crowdsourcing annotations may be hard or not possible, *e.g.*, for precise metric depth ground-truth we need on-site collection with specialized devices. Hence, target domain (*i.e.*, the distribution in which we aim to have good performance) annotation may not be economically feasible for most use cases. It is estimated [63] that the market for data labelling solutions was $1.7B in 2019, projected to grow to $4.1B by 2021, while data engineering tasks consume 80% of the time in AI projects. We need to find approaches more cost-efficient than data annotation.

Reducing the Need for Annotations. All these results beg the following question: *Can we achieve good results even if we do not have (enough) annotated data?* Several directions have tackled this question. For example, the self-supervised direction assumes access only to a non-annotated dataset to train good feature representations [47, 120, 100], although it still needs annotations for downstream

tasks (*e.g.*, segmentation or detection). In the semi-supervised case, non-annotated data is mixed with a subset of annotated data [310, 23], meaning a subset of annotations in the real domain is still needed, which can be hard to acquire. The domain generalization setting [229] assumes no access to the target distribution and leverages other annotated datasets to train networks capable of performing well on unseen data. However, we argue that obtaining non-annotated target data is usually manageable and can further improve the results obtained by domain generalization methods [86, 41]. The combination of a target dataset with other datasets during training is leveraged in the main setting tackled in this thesis, **Domain Adaptation (DA)**.

What is Domain Adaptation? Domain Adaptation aims to obtain good performance in the target domain by using data already annotated in a different domain, usually called the source domain. In this context, a domain refers to a specific data distribution, where two datasets collected under different settings (*e.g.*, cameras, location or weather) form two different domains. Specifically, we will mostly focus on a specific subset of DA, Unsupervised Domain Adaptation (UDA), which assumes there is no ground-truth available in the target domain. The machine learning community has spent significant resources to annotate large real datasets (or creating fully-annotated synthetic datasets), thus Unsupervised Domain Adaptation aims to leverage the already available data as a source domain without needing to annotate any extra data. Due to the bias existing in the different vision datasets [327], a network trained solely in a given source dataset will learn from a limited distribution and will have trouble generalizing to our target dataset. Applying UDA techniques can reduce the impact of this distribution mismatch. For instance, following the presented example of SVHN→MNIST, employing a UDA method we can achieve an accuracy of 99.3% [89] improving, without using any extra annotations, upon the accuracy of 71.7% obtained with only naive source data training.

Synthetic Data. Reusing real domain datasets is one of the motivations of using DA methods, but another focus of DA approaches is using synthetic data (*e.g.*, from a simulator) to train a model capable of obtaining good performance in a real domain. Even though some works are advocating for data-driven or neural simulators [160, 159], most current Computer Vision simulators are built by artists and engineers using a game engine. A close to photorealistic simulator with significant variability in 3D models, textures, situations or illumination conditions can take a long time to develop. However,



Figure 1.2: Synthetic data advantages. Synthetic data can produce a vast amount of detailed multimodal data (left image [131]), can safely simulate long-tail events hard to encounter in real life (center image [313]), and is privacy-friendly (right image, faces are synthetic [369]).

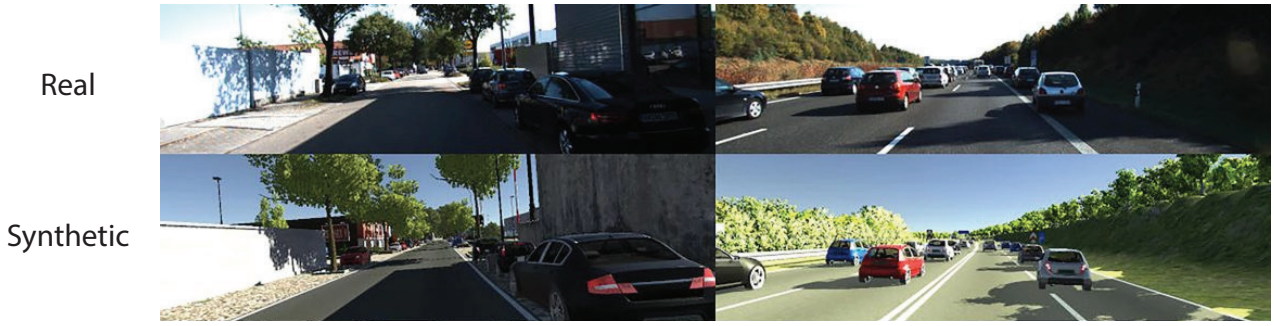


Figure 1.3: Real (KITTI [97]) and Synthetic (Virtual KITTI [92]) examples showing a visual domain gap. The real and synthetic images represent the same semantic scene, however the differences in illumination and textures, among others, hamper the generalization on the real data of a model trained with synthetic data.

as illustrated in Figure 1.2, once a simulator is developed, it can be easy to generate a large amount of synthetic data. The generated synthetic data can be diverse as the conditions can be easily randomized, where the simulator is capable of also generating rare events that are uncommon in real data. Furthermore, synthetic data from a simulator is privacy compliant by default. Several companies are working on synthetic data generation such as NVIDIA [241], Apple [272] or Facebook [297, 320]. It is projected that by 2030 most of the generated data for training will be synthetic [241, 261]. Even though in some cases synthetic data can be used directly to increase the performance of a model with limited data [237], models trained in synthetic data usually do not behave well when deployed directly in a real scenario due to the domain shift, as shown in Figure 1.3, which is why DA methods are needed for unlocking the potential of synthetic data.

Summary. The machine learning community needs to keep investigating methods to reduce the burden of data annotation. One of the main trends in that direction is Domain Adaptation. Even though ideally a model should be able to generalize to new datasets without seeing any target domain data,

current methods trained in a standard supervised manner have issues when tested on target data with different distributions. Hence, we believe Domain Adaptation, and specifically Unsupervised Domain Adaptation, is a good middle-ground before the development of fully generalizable methods since obtaining non-annotated data is in most cases of acceptable difficulty. Being able to quickly and reliably finetune a model in hard-to-annotate tasks (*e.g.*, depth completion, object detection, or semantic segmentation) by using a set of unlabeled target images can unlock a large number of applications, along with reducing the cost of deploying a machine learning model.

1.1 Contributions and Outline

Many of the current Domain Adaptation approaches use combinations of several well-known techniques, which we present in **Chapter 2**. In this thesis, we tackle different semantic and 3D tasks under Domain Adaptation settings. For these semantic and 3D tasks, we aim to study the effectiveness of techniques that have been widely researched in a classification context. We thus first investigate a closely-related task to classification, *i.e.*, object detection, which is a harder task to annotate compared to image classification (see Figure 1.1). We then transition to 3D-related tasks, which can benefit from developing successful DA methods but have been under-investigated by the literature. Specifically, for the second and third tasks we leverage the readily available dense and noise-free ground-truth in simulators, which is difficult to obtain in real data due to limitations of the depth acquisition methods, to investigate depth estimation and depth completion. Lastly, we make use of the multi-camera projection techniques used in our depth-related tasks (*i.e.*, depth estimation and depth completion) to generate synthetic views that are used in our fourth studied task, *i.e.*, camera pose regression. Each of our proposed methods addresses a different research question, however, there is a common theme shared in all chapters: the investigation of DA techniques that are effective for the proposed tasks. Even though the tasks and the settings differ, the DA approaches we propose are inspired by common DA methods, thus share similarities at their core. We first list the high-level contributions for each chapter related to our Domain Adaptation motivations. After the individual chapter contributions, we also expose the global contributions and observations of this thesis.

Chapter 3 - Domain Adaptation for Object Detection via Style Consistency [208]. Many Domain Adaptation methods assume a consistent image style in the target domain, however in some cases that assumption can be challenged. We, therefore, address the following question, *How can we adapt well-known Domain Adaptation methods to a multi-style target dataset?*. In this context, *style* refers to low-level changes, such as variations of color distribution or use of different texture to represent the same semantic instances (see Section 2.1.1 for an expanded definition of *style* and *content*). This question is researched for the task of *object detection*, which presents different challenges compared to object classification, *e.g.*, locality of predictions and unknown instances per image. Considering these challenges, and our research question, we introduce the following contributions in Chapter 3:

- We propose the use of an image adaptation method capable of transferring to individual styles in a setting where the target domain can have multiple visual styles. The results show that representation of multiple target domain styles leads to a better adaptation.
- High-level semantic features for object detection should be invariant to the style. We thus introduce feature consistency between the original and the style-transferred image to increase the robustness of the detector to low-level variations.
- We propose a pseudo-label strategy that is crafted for object detectors. Our pseudo-label strategy introduces reliable positive and negative examples from the target domain into training.

Chapter 4 - DESC: Domain Adaptation for Depth Estimation via Semantic Consistency [209]. Consistency-based techniques and pseudo-labels are key in Domain Adaptation, and the combination of modalities can provide extra cues for a successful adaptation. Noticing that humans use semantic information to guide any depth inference, we ask *Can we exploit the natural relationship between depth and semantics to guide the adaptation of a depth estimation network?*. We explore this question in a *monocular depth estimation* context, mainly without any self-supervision (*e.g.*, video frames or stereo cameras). On that note, Chapter 4 presents the following contributions:

- We introduce a novel consistency constraint with predictions from a second model trained on high-level semantics and low-level edge maps.

- We propose an approach to form depth pseudo-labels in the target domain by using object size priors, which are learned in an instance-based manner in the annotated source domain.

Chapter 5 - Project to Adapt: Domain Adaptation for Depth Completion from Noisy and Sparse Sensor Data [207]. *Depth completion* uses two input modalities (sparse depth and RGB), where both present a different domain gap in a Domain Adaptation context. We thus ask *How important is the adaptation of the RGB and input depth modalities?*. The contributions in Chapter 5 are:

- We use a simulator to collect and release a multi-camera synthetic dataset mimicking the set-up of a well-known real dataset, KITTI [97].
- We provide an explanation for the reduced performance on real data when naively training on synthetic data. Based on this observation, we develop an approach for geometric and data-driven sensor mimicking aimed to generate realistic input sparse depth.
- We show that RGB adaptation is secondary compared to input depth adaptation.
- Naive pseudo-labelling (*i.e.*, using input depth as supervision) limits the achievable performance. We thus propose a method for input data filtering to boost the performance.

Chapter 6 - Improving Camera Pose Regression via Synthetic Views [237]. The use of synthetic data is an important motivation for Domain Adaptation pipelines. In the task of *camera pose estimation* the use of synthetic views augments the coverage of a scene, however, the synthesized views contain many noise artefacts. We thus ask two questions, first *Is the lower performance of CNN-regression methods compared to classical methods (partly) due to data issues (*e.g.*, bias or amount of data)?* and the subsequent question *Can a CNN-based pose predictor learn from synthetic views with noise artifacts?*. The work presented in Chapter 6 was done jointly with another author, hence the contributions are shared. While we state all of the contributions of this work, we also clearly mark the contributions claimed in this thesis.

- We propose to use synthetic data to reduce the gap between structure-based and regression methods, showing that regression networks may compete with 3D methods with less overhead.

- Our approach is the first regression method to improve upon image retrieval on the largest and most complex *Street* scene from [154]. We analyze the reasons why regression methods do not work directly on this scene and propose a solution.

- We propose a pipeline for training camera pose regression methods based on a probability-driven selection and generation of synthetic views. The generation of synthetic views is done using multi-image projection and a combination of sparse but up-to-scale ground-truth depths with densely predicted scale-agnostic depths to improve the quality of synthetic images. (contribution of this thesis)

- We show that a global regression network is robust to the image artifacts, being able to boost the results of the pipeline with synthetic views of variable quality. (contribution of this thesis)

- We show that a transformer [334] based architecture is more suitable for inferring relative pose regression, outperforming the standard MLP approaches adopted by all previous methods.

Overall Contributions. Throughout this thesis, we investigate the use of standard Domain Adaptation techniques and assess their usefulness for different tasks. On that note, we find the following overall trends regarding DA techniques:

- Data adaptation is a sensible first step for Domain Adaptation tasks, usually through the use of adversarial image techniques (see Chapter 4). However, for increased performance we can also leverage the knowledge of the target task to use adaptation methods that target a specific issue (*e.g.*, style-transfer for multi-style adaptation in Chapter 3). Furthermore, we can directly adapt the synthetic data in the generation pipeline to avoid the use of any learning-based adaptation on top (*e.g.*, collection of multi-camera data to generate projection artifacts in Chapter 5, or joint probability pose selection with multi-image projection in Chapter 6).

- Introducing consistency in the target domain constraints the learning process and usually leads to increasing the target domain performance. The consistency term employed can be specific to the task we aim to solve and can be applied in several parts of the network (*e.g.*, high-level feature-consistency in Chapter 3 and output prediction semantic consistency in Chapter 4).

- Target domain pseudo-labels can be beneficial for Domain Adaptation tasks, however, we need to develop methods to reduce the noise of the labels to avoid hurting the target performance (*e.g.*, confidence filtering in Chapter 3 and object thickness filter in Chapter 5).

Finally, in **Chapter 7** we present a summary of our observations and limitations of the presented methods, along with future trends in the Domain Adaptation.

1.2 Publications

The main results presented in this thesis correspond to the following publications:

- A. Lopez-Rodriguez, Krystian Mikolajczyk. Domain Adaptation for Object Detection via Style Consistency. Proceedings of the British Machine Vision Conference (BMVC). September 2019.

- A. Lopez-Rodriguez, Krystian Mikolajczyk. DESC: Domain Adaptation for Depth Estimation via Semantic Consistency. Proceedings of the British Machine Vision Conference (BMVC). September 2020. *Oral Presentation.*

- A. Lopez-Rodriguez, Benjamin Busam, Krystian Mikolajczyk. Project to Adapt: Domain Adaptation for Depth Completion from Noisy and Sparse Sensor Data. Proceedings of the Asian Conference on Computer Vision (ACCV). December 2020. *Oral Presentation and Best Student Paper.*

- Tony Ng*, A. Lopez-Rodriguez*, Vasileios Balntas, Krystian Mikolajczyk. Reaassesing the limitations of CNN camera pose regression. Submitted March 2021. * equal contribution, order decided by (virtual) coin toss. Preprint

- A. Lopez-Rodriguez, Benjamin Busam, Krystian Mikolajczyk. Project to Adapt: Domain Adaptation for Depth Completion from Noisy and Sparse Sensor Data. International Journal of Computer Vision (IJCV) ACCV20 Special Issue. Submitted April 2021.

- A. Lopez-Rodriguez, Krystian Mikolajczyk. DESC: Domain Adaptation for Depth Estimation via Semantic Consistency. International Journal of Computer Vision (IJCV) BMVC20 Special Issue. Submitted August 2021.

Chapter 2

Background

We now introduce the relevant concepts, datasets and background for the work presented in the following chapters. We first discuss the current trends in Domain Adaptation (DA). We then introduce in a DA context each of the tasks tackled in this thesis: object detection, depth estimation, depth completion and camera pose estimation.

2.1 Domain Adaptation

Past works developed some theoretical grounds for Domain Adaptation [20, 22, 217, 21], mostly in a pre-Deep Learning context. Although current state-of-the-art DA approaches share insights with the DA theory proposed in [20, 21], there is also a gap between that prevalent DA theory and the currently proposed ad-hoc algorithms [3], with some past works trying to bridge this gap [370, 263, 3]. In this chapter, we mostly focus on presenting the practical techniques used currently that commonly yield positive experimental results.

Problem Setting. Domain Adaptation aims to leverage data from both a source dataset $\mathcal{S}$ and a target domain $\mathcal{T}$, to have good performance in $\mathcal{T}$. Furthermore, DA also assumes that the source domain is annotated for the same task we aim to solve in $\mathcal{T}$. Specifically, our work mainly tackles an Unsupervised Domain Adaptation (UDA) setting[1], which assumes that the target domain is not

[1]Except for the pose estimation work in Chapter 6, where we mix annotated synthetic data with annotated real data.

annotated. In our settings, we also assume that we only have a single domain in both $\mathcal{S}$ and $\mathcal{T}$. For semantic-related tasks (*i.e.*, our object detection work in Chapter 3) it is also assumed that both $\mathcal{S}$ and $\mathcal{T}$ share the same classes. This assumption of class sharing is referred to as closed-set setting.

Alternative Domain Adaptation Settings. Even though we do not explore them in our work, current Domain Adaptation trends also investigate settings that differ from the single-source and single-target closed-set domain setting. A non-exhaustive list of variations are:

- **Multi-source DA:** multiple source domains are used during training [249, 375]
- **Partial DA:** the target domain contains a subset of the source domain classes [38, 37].
- **Open Set DA:** target and source domain do not share the same classes[243, 287].
- **Universal DA:** no prior label knowledge in the source and target data [284].

The list of variations is constantly expanding, aiming in most cases for less-restrictive (*i.e.*, less prior knowledge) problem definitions to allow for more flexible use of both the annotated data and the non-annotated data.

Domain Shift. Datasets contain differences that make them easily recognizable [327]. These differences, or distribution changes, between two different datasets are referred to as domain shift. Deep Learning approaches are sensitive to changes in the input distribution (*e.g.*, while lower layers learn more general and transferable features, higher layers learn domain-specific features [203, 361]), hence domain shifts between training and testing data tend to heavily decrease the performance of the network [93, 252, 115]. In vision datasets, a domain shift can refer to low-level shifts, such as changes in illumination, contrast, saturation, color palette, or to more higher-level shifts such as differences in backgrounds or shape of objects. For example, in an autonomous driving setting, different domains can arise due to night/day differences [273], weather condition changes [288], or differences in country of collection [235] leading to differences in road or traffic signals appearances or density of traffic, among others. In 3D-related tasks, there are also domain differences due to changes in the camera intrinsics [84] or camera poses used [377].

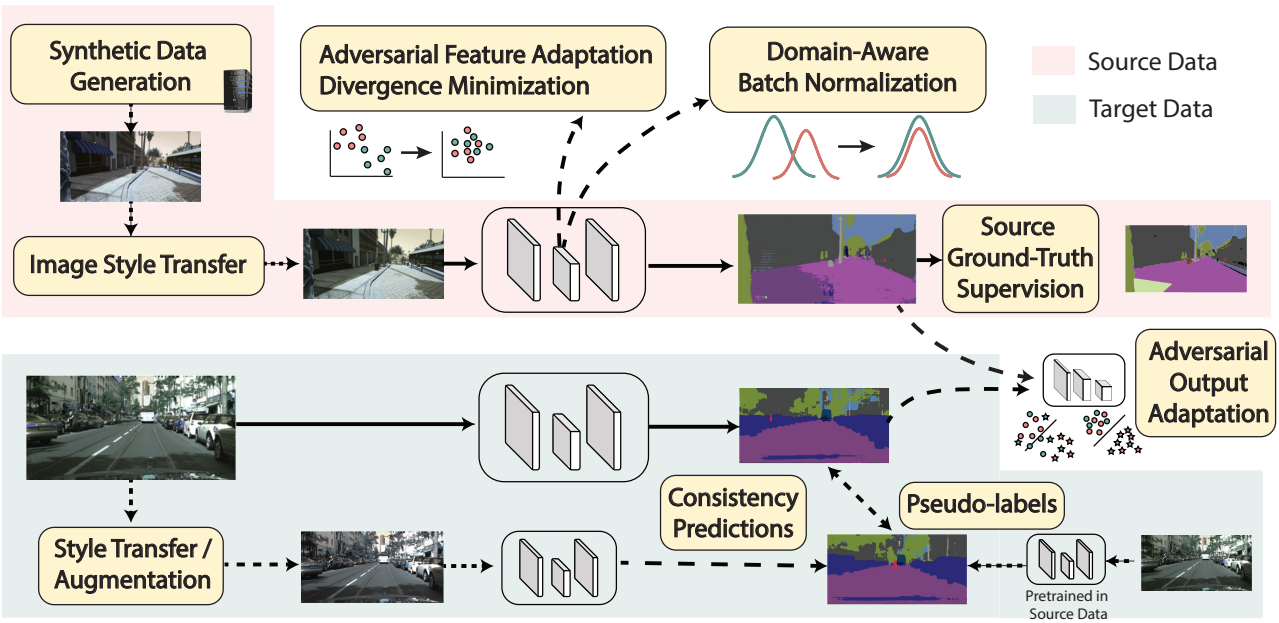


Figure 2.1: Overview of Domain Adaptation techniques. The yellow blocks refer to techniques that are usually applied in a Domain Adaptation work, which we describe in this chapter. Dashed lines show possible techniques applied, whereas solid arrows refer to the standard modules used in Unsupervised Domain Adaptation works. RGB and segmentation examples are taken from [53].

2.1.1 Current Domain Adaptation Techniques

Due to the domain shift, Deep Learning models trained on a source dataset may not work well in the target dataset. A large number of works have developed Domain Adaptation approaches aiming to bridge the gap between the performance in the source and the target dataset. Classical DA literature adapted precomputed features with shallow DA methods [107, 205], whereas current works train jointly a Deep Learning model with a DA method (in some cases, this is referred to as Deep Domain Adaptation). We now present the main techniques used currently in Computer Vision Deep Domain Adaptation works (focusing on Unsupervised Domain Adaptation), which are depicted in Figure 2.1.

Source Data Supervision. In DA techniques, a training batch commonly includes both source and target examples with a 50/50 split [209, 207], as alternating batches of source and target data may hurt the performance [116]. To train the model, the network is supervised with source data ground-truth, whereas extra techniques are used for the non-annotated target data (*e.g.*, pseudo-labels or matching source and target feature distributions). Some works may perform first a pretraining step using only the fully annotated source data, and then a second step of joint training with both source and target data [285, 207].

Divergence-based methods minimize a divergence loss to align the statistics of the source and target features. Some of the most notable methods are Maximum Mean Discrepancy (MMD) [111, 16], widely used in classical methods, Correlation Alignment (CORAL) [316, 317], or using an optimal transportation method [68, 67, 70, 176]. MMD represents the distances between distributions as the distance between the mean of the embeddings, CORAL aligns the second-order statistics (correlation) of the source and the target domain, and the optimal transportation methods minimize a Wasserstein discrepancy to produce an optimal transport plan between source and target pairs.

Adversarial Alignment, like divergence minimization methods, aims to align the distributions of the source and target features. In this case, the alignment is performed in an adversarial manner [93], where different variations have been commonly used in recent works [93, 330, 368, 335, 362]. Adversarial alignment takes intermediate feature vectors extracted from source and target domain data, and places a domain discriminator on top of these features to minimize the standard minimax adversarial loss [108]. A variation of the two-step minimax loss is using a gradient-reversal variant [93], which is popular in semantic works [127, 51, 286]. Due to the minimax loss, the feature extractor, which aims to fool the domain classifier, aligns the source and target features. The alignment is originally performed without any notion of class or label in the target domain, which may result in a drop in performance due to negative transfer, leading some works to tackle this drawback [248, 204]. Furthermore, the alignment also assumes that the whole feature vector needs to be adapted, *i.e.*, the whole vector should be *domain agnostic*. A body of work explicitly challenges this assumption by disentangling the components that are *domain specific* from those that are *domain agnostic* [25, 43, 250].

Image Adaptation aims to transform the visual appearance of the source images to follow the target distribution. Image adaptation increases the target domain performance in several tasks, including regression tasks (*e.g.*, depth estimation), where other techniques are less effective. Input adaptation methods mainly reduce the gap in textures, illumination, color palette or contrast while preserving high-level semantic content [382, 106], hence source domain labels can still be used on the transformed images. Image adaptation is usually combined with other DA techniques for increased performance [53, 379, 209]. We highlight two main trends when adapting the input image. The first trend is performing *style transfer* [346, 208, 157]. The second one, which is more widely used, is *adversarial*

image adaptation [126, 53, 379]. We now briefly introduce both trends as they will be used in our proposed methods.

- **Style Transfer.** We refer to style-transfer when we output an image with the semantic information of a specific *content* image, while using the *style* of another individual image. Even though the definition of style is somewhat ambiguous, the style is usually related to the texture information and global color distribution [96, 80] of an image. Conversely, content refers to the objects and general semantic information present in the image. In artistic style-transfer settings, the transfer of texture information can also be coupled with some local modifications of the shape of the object (*e.g.*, transferring a *cubist* style to an image of a real face may alter the shape of the different elements of the face) [96]. However, in realistic image settings, the shape of the semantic objects needs to be strongly preserved in the style-transferred images, and thus modifications of the style mostly refer to color statistics, illumination or weather information [360]. In a Domain Adaptation setting, the style image is usually sampled from the target dataset $x_{\mathcal{T}}$, whereas the content image comes from the source dataset $x_{\mathcal{S}}$, as we aim to leverage source data annotations, which are usually related to the content. An initial Deep Learning style transfer method was proposed by Gatys *et al.* [96], which used a costly iterative optimization process, while further extensions used a feed-forward approach [148, 332, 80, 134, 188, 360], which is less costly at inference time. Some DA methods leverage these Deep Learning style-transfer approaches [346, 157], while other DA methods use simpler style-transfer methods as for example transferring image statistics [311, 141, 2].

- **Adversarial Image Adaptation** aligns the source image distribution to the target image distribution using methods based on Generative Adversarial Networks (GAN). Contrary to *style transfer*, adversarial image adaptation works on a distribution level to obtain $x_{\mathcal{S}\to\mathcal{T}}$ from $x_{\mathcal{S}}$. Furthermore, there is no guarantee that individual target styles are represented in the adaptation. Standard adversarial image translation approaches only generate one mapping $x_{\mathcal{S}\to\mathcal{T}}$ from each source example, although some works have proposed extensions to remove this limitation [6, 135]. DA literature use commonly CycleGAN-based [382, 126] methods, which does not need paired data. In a standard conditional adversarial network pipeline [142], after us-

ing a generator G to transform $G(x_{\mathcal{S}})$ to $x_{\mathcal{S}\to\mathcal{T}}$, we use a discriminator on top of the $x_{\mathcal{S}\to\mathcal{T}}$ and $x_{\mathcal{T}}$ examples. However, with unpaired data, there is no guarantee that $x_{\mathcal{S}\to\mathcal{T}}$ will have a similar content to $x_{\mathcal{S}}$. On that note, CycleGAN adds a cycle consistency term that uses the transformations $x_{\mathcal{S}} \to x_{\mathcal{S}\to\mathcal{T}} \to x'_{\mathcal{S}}$ to enforce consistency between $x_{\mathcal{S}}$ and $x'_{\mathcal{S}}$ (a similar cycle is repeated for $x_{\mathcal{T}} \to x_{\mathcal{T}\to\mathcal{S}} \to x'_{\mathcal{T}}$). Some DA approaches remove this cycle consistency term to reduce complexity, and instead ensure the semantic consistency of $x_{\mathcal{S}}$ and $x_{\mathcal{S}\to\mathcal{T}}$ via a task loss [379, 329]. Adversarial image adaptation is currently the most widely used image adaptation approach due to the high-quality translation that can be achieved [379, 126, 140].

Pseudo-labelling, or self-training, is another widespread technique. In a pseudo-labelling approach, we use a classifier trained in the source data to label (or *pseudo-label* in DA nomenclature) new examples from the unlabeled target domain. After the examples have been pseudo-labeled, we include them in our training set and finetune our classifier with them [386, 368]. Training with target pseudo-labels and source labels enforces the classifier boundaries to pass through low-density regions, *i.e.*, regions with a low number of source or target examples, which follows the assumption that features are clustered based on their label [177]. Pseudo-labels can also be used successfully in problems that output a map, such as semantic segmentation [386, 385]. In a similar direction to pseudo-labelling, some works use entropy minimization in the target domain [206, 336]. Entropy minimization is commonly used in semi-supervised learning [110, 226], and the rationale is, as when using pseudo-labels, to move the classifier boundaries further away from the data points by forcing the target domain probability output to be more *peaky*. Pseudo-labeling can be combined with curriculum learning, which is especially used in image classification [285, 368] and semantic segmentation works [386, 242, 193]. In a curriculum learning approach, the target data is progressively pseudo-labeled from easy to hard using the prediction confidence [285, 242], while some works tackle the bias occurring from pseudo-labelling mainly easy-to-transfer instances [368, 386].

Consistency. The learning process in the target domain can also be constrained by using consistency terms. In a UDA setting, our target domain is unlabeled, however, we expect the predictions (or the features in some cases) for augmented or perturbed versions of a target image to be consistent with the predictions of the initial image. On that note, some works force consistency between the

predictions of the original image and the predictions of a style-transfer [53] or augmented [221, 276] image, whereas others [168] use consistency between multiple tasks. Some authors force consistency on a feature level [365, 208] instead of on an output level, which is more similar to the contrastive approaches that are popular in self-supervision tasks [47]. Self-ensembling in the form of a mean teacher [323], where the teacher is a network built using an exponentially moving average of the student weights, can also be combined with consistency terms and augmented images [89, 72, 35].

Batch Normalization with domain-specific parameters can also be used to generate domain invariant features by aligning source and target distributions. AdaBN [190, 189] updated the batch normalization layers with the running statistics of the target dataset in a post-processing stage, and further work integrated the domain-specific batch normalization layers in the training pipeline [42, 215, 219], or replaced batch normalization with a whitening layer [276]. In the spirit of having domain-specific parameters, some works propose different branches [206] for each domain, where other works propose to untie the weights, either allowing them to vary freely [330], regularizing them to force target and source weights to be close [278], or using a residual term to transfer source to target weights [279].

Output Adaptation. Tasks with structured outputs, such as semantic segmentation and depth estimation, may also benefit from adapting the target domain output map. Similarly to data and feature adversarial adaptation, we can use a discriminator, along an adversarial loss term, to force the target domain output maps to follow a similar distribution to the source domain output maps. Output adversarial adaptation has shown to be more beneficial in classification settings compared to regression settings, leading to, *e.g.*, a larger number of works applying it in a semantic segmentation [337, 328, 336] context compared to a depth estimation [233] task.

Synthetic Data Generation is tackled by several works, usually via the use of a simulator, aiming to generate labeled data closer to the target domain data. There are several trends to improve the data generation step. The first trend aims to increase the visual quality of the rendered data by using more detailed 3D models, better textures or new illumination techniques like raycasting, which is fueled by the advancements in gaming and movie industries [92, 32, 185]. A second direction aims to match the synthetic data to the real data at a higher level by learning to adapt *e.g.*, the number of objects or camera pose in a simulator [271, 151, 73]. A third recent direction generates the synthetic data using

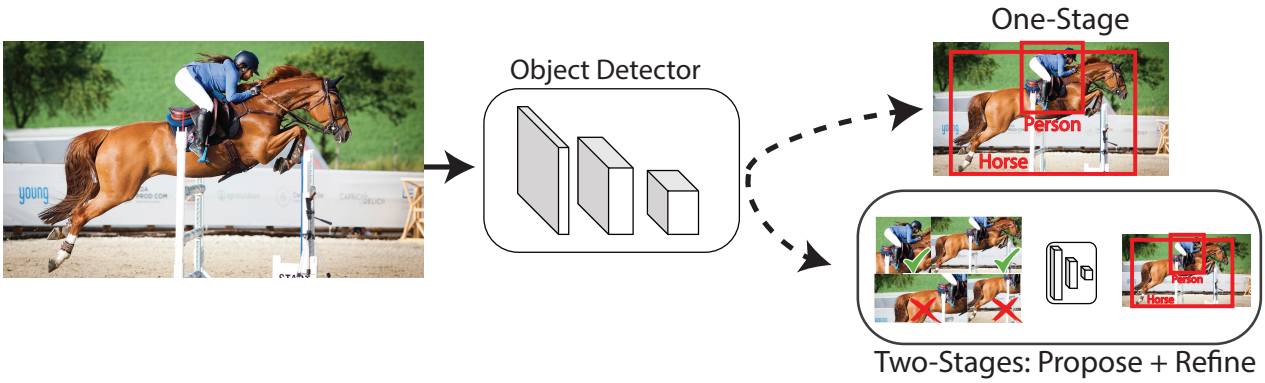


Figure 2.2: Object detection problem. Given an image, we aim to retrieve a bounding box and class for all instances of objects (usually from a predetermined set of classes) in an image. Deep Learning object detectors can be classified into two-stages or one-stage detectors depending on whether they use a refinement step with feature pooling.

Deep Learning models, either by sampling examples from a generative model [369] or by directly building a controllable neural simulator [160, 159].

Now that we presented the main techniques used in Domain Adaptation settings, we briefly introduce the different tasks tackled in this thesis within the context of Domain Adaptation: object detection, monocular depth estimation, depth completion and camera pose regression. We aim to give an overview of the main ideas for each of the tasks in order to understand better the work in Chapter 3, Chapter 4, Chapter 5 and Chapter 6.

2.2 Object Detection

Problem Setting. As depicted in Figure 2.2, object detection aims to predict the objects (among a set of predefined categories) in an image, outputting both the class and location (using a bounding box) of the object. Unlike an image classification setting, object detection has an unknown number of instances in an image. Annotation-wise, each image needs a variable number of class and bounding box labels, where the objects can be occluded or superimposed to each other. Compared to image classification, obtaining annotations in a semi-automatic manner via user-provided captions is more difficult due to the unknown object locations and number of instances in an image [358]. These annotation challenges lead to standard object detection datasets being usually smaller than image

classification datasets.

Deep Learning Methods. Object detection performance has been largely improved by Deep Learning methods [102], where the methods can be separated into two-stage networks and one-stage networks. *Two-stages networks* use first a region proposal step to filter out low-confidence regions and output plausible object proposals, and a classifier is then applied on top of the proposed regions. Some well-known two-stage detectors are R-CNN [102] and Fast R-CNN [101], and Faster R-CNN [265], where the latter added a region proposal network to make the model end-to-end trainable. *One-stage networks* do not have a region proposal step, leading to more efficient inference. Examples of one-stage detectors are YOLO [264] and SSD [201].

Domain Adaptation Trends. Object detection presents a locality challenge (*i.e.*, the object is detected in an image region) and a variable number of instances per image. To take into account the local nature of the prediction, the detectors need to be adapted at multiple levels [286]. Furthermore, the unknown number of instances in an image affects the strategies based on pseudo-labelling [208, 277]. Several object detection Domain Adaptation methods use Faster R-CNN [35, 51, 286], which is a two-stage detector, and adapt also the region proposal network [35, 286], whereas other methods use one-stage networks as a base model, mainly SSD [208, 140]. As for observed DA trends, *adversarial feature adaptation* is widely used [124, 51, 384, 348, 304, 45, 351, 286, 315]. Adaptation of both the instance-level features and the image-level features was done in [51] and expanded in [348, 124, 45] by adding adversarial adaptation for the earlier layers, and in [351] by weighting the adversarial feature alignment loss with an image-level multi-label classifier. Another work [384] clustered the different regions proposed by the Faster R-CNN region proposal module to obtain local feature vectors that are adversarially adapted. However, scenes may differ (*e.g.*, different combination of objects) between domains, thus Saito *et al.* [286] used a focal loss [194] to adapt the high-level features of only the similar scenes between the two domains. Instead of using standard adversarial feature adaptation, Su *et al.* [315] applied a conditional adaptation aimed at only adapting the domain information. *Image adaptation* is another trend that has found success in object detection [140, 304, 161, 329]. In [140] CycleGAN was used to perform a pixel-level adaptation of source images to the target domain, where the adapted images were then used to train a detector. On that note, Tzeng *et al.* [329] integrated

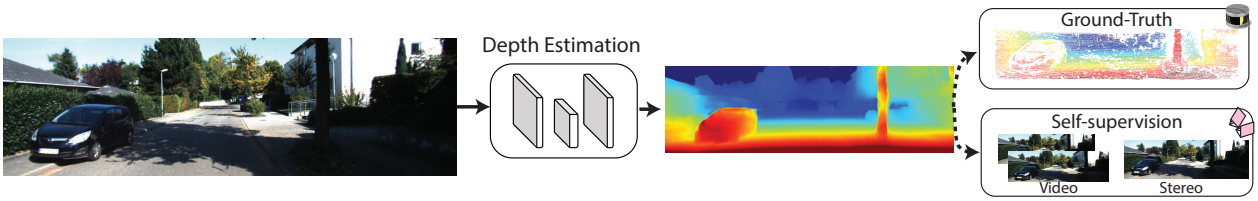


Figure 2.3: Monocular depth estimation problem. For a given individual RGB image, we aim to estimate a dense depth map, which can be in a known (*e.g.*, metric) or unknown scale depending on the setting. Supervision for this task can be in the form of a given ground-truth, or leveraging video or stereo sequences during training (which we assume are not available at inference time).

an adversarial translation module in the object detection end-to-end training to remove the need for the computationally expensive cycle consistency. In a multi-style approach, Kim *et al.* [161] tackled the one-to-one mapping limitation of standard adversarial-based image translation modules by generating multiple translated images per source example. To do so, in [161] several adversarial image generators were trained with different reconstruction loss terms. Self-training using *pseudo-labels* has also proven beneficial in past works [277, 158, 140], where some works specifically tackled the negative mining used in object detection [158], others assumed videos and refined the pseudo-labels via trackers [277], and others [155] focused on the robustness of the model to noise in the pseudo-labels. Finally, *consistency* terms have also been beneficial for this task, in this case with a mean teacher network [323] as used in [35] and the unbiased modification in [72].

2.3 Monocular Depth Estimation

Problem Setting. Depth estimation refers to predicting a dense depth map from RGB images, which is crucial for several tasks (*e.g.*, robotics, augmented reality, virtual reality). As shown in Figure 2.3, in our work we focus on the monocular case, where we estimate the depth out of a single RGB image without using any stereo information during inference. Regarding depth annotations, scale-less depth can be collected via a MultiView Stereo (further explained in Section 2.5), which needs a database of multiple images around the same scene and generates an output depth map with a non-negligible amount of noise [192]. To obtain a precise metric scale depth, we need an on-site collection with special devices (*e.g.*, LiDAR [97] or Structured Light [307] sensors). Accurate sensors with long ranges, which are needed for outdoor data, are costly, which makes annotation expensive.

Deep Learning Methods. RGB based depth estimation has a long history [300, 173, 325]. Recent approaches using Deep Learning are capable of estimating a high-quality pixel-wise depth from a static image [170, 105, 114, 192, 104, 262]. Different models have been proposed for this task, where most of them use an encoder-decoder [170, 104, 178] approach. Although initial approaches used a custom network [82], it is common to leverage a well-known backbone (*e.g.*, ResNet [122] models [104]), which can be pretrained in ImageNet [71, 104, 5] for increased performance. Even though the depth predictions from a model are in some cases of high-quality, monocular depth estimation is an ill-posed problem, as a single 2D image can be produced by several 3D scenes. Humans are capable of estimating plausible depths given a single image in a novel scene using cues such as perspective, illumination or scale of known objects [74], however, the cues used in a Deep Learning model are still being investigated. For example, in [74] it was found that in the driving dataset KITTI [97], monocular depth estimation models leveraged the vertical object position for the prediction, and Ranft *et al.* [262] found that the model was biased to predict lower parts of the image to be closer.

Self-Supervision. A large body of literature has aimed to remove the need for real ground-truth data via the use of self-supervision during training [105, 104, 359]. In a self-supervised approach, we assume we have either a video or stereo sequence used to guide our network via photometric consistency, which compares a reference RGB image x_2 with a projected RGB image $x_{1\to2}$, where x_1 and x_2 can be two frames in a video or also two different images from a stereo setup, and $x_{1\to2}$ is the projected image x_1 to the camera pose of x_2. A photometric loss, like MSE or SSIM, is then used to compare the pixel values of $x_{1\to2}$ and x_2. Self-supervision presents a wide range of issues, namely inaccuracies on non-Lambertian surfaces or dynamic objects [305], calibration problems for stereo rigs or errors in simultaneous pose and depth estimation for video-based supervision [104]. The combination of self-supervised methods and DA methods with synthetic data is a promising direction to alleviate these issues.

Domain Adaptation Trends. Some DA works assumed no target domain self-supervision [379, 233, 10], whereas others combined self-supervision with DA for increased performance [374, 54, 116]. Regarding DA techniques, *image adaptation* has been widely used to decrease the domain gap [10, 374, 379, 256, 54]. In [10] a cycle-consistent image-transfer approach was used with added inference-

time image translation. T^{2}Net [379] removed the cycle-consistency loss, reducing the translation complexity, and did not use inference-time translation. Several works have built upon T^{2}Net [256, 54, 374]. The Geometry-Aware Symmetric Domain Adaptation (GASDA) method [374] added stereo photometric guidance, along with depth consistency between the original and the style-transferred target images. SharinGAN [256] transformed both domain images into a shared domain with a learned generator, which improved results but increased complexity at test time. S^{3}Net [54] leveraged video self-supervision and semantic maps to impose extra constraints in the translation step, namely, multi-frame photometric consistency and semantic consistency. In a semi-supervised Domain Adaptation task, Attend-Remove-Complete (ARC) [378] argued that real-domain clutter is one of the reasons for the synthetic-to-real domain gap, and used image adaptation to remove it. In a domain generalization context, S2R-DepthNet [49] transformed the input image to a structural edges map thus removing unnecessary information (*e.g.*, textures), which allowed it to use only synthetic data to train a model that generalizes to unseen real data. AdaDepth [233], instead of using image translation, employed *adversarial feature and output alignment* between the source and target domain, along with feature *consistency* to avoid mode collapse. MonoDEVSNet [116] also used adversarial feature alignment, while leveraging video self-supervision simultaneously. In depth estimation (and other 3D tasks), changes in camera intrinsics or poses also affect the domain gap. Including the camera parameters information via a special layer [84], or predicting the intrinsics during training [109], helps when dealing with variable intrinsics. Furthermore, past work [377] indicated that the training poses used affect the depth estimation adaptation.

2.4 Depth Completion

Problem Setting. Depth completion aims to output a dense depth given a sparse input depth. The input sparse depth is usually combined with a dense RGB image that serves as guidance for the completion method, although depth completion based only on the sparse input is also possible [333, 331]. The sparsity pattern of the input depth varies depending on the sensor used, which affects the accuracy of the final depth map [350]. Two of the most used type of sensors to obtain depth, LiDAR and Structured Light, have different sampling patterns as shown in Figure 2.4. The original version of

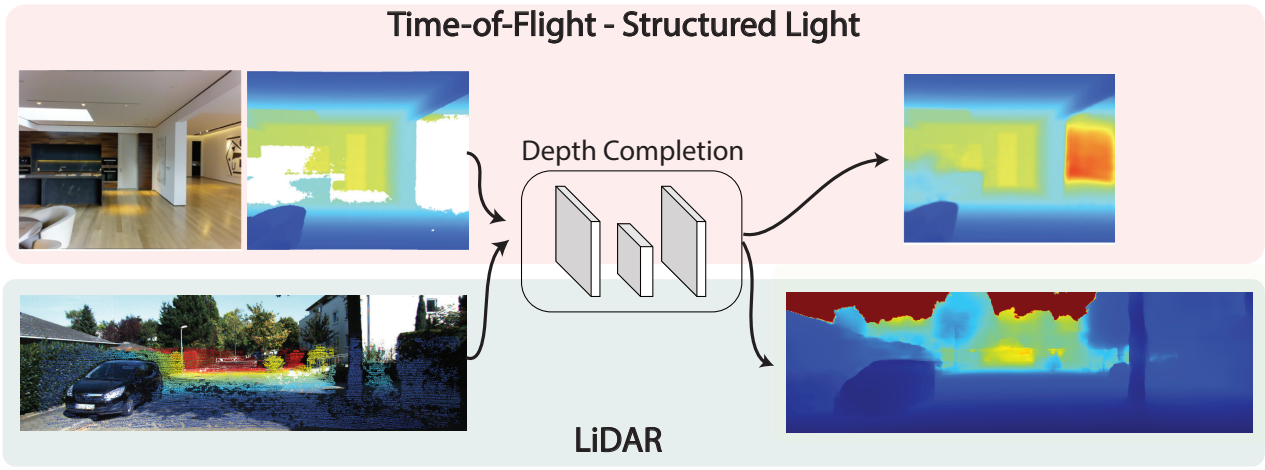


Figure 2.4: Depth completion problem. Given a sparse depth, we aim to output a fully dense depth map. The depth sparsity pattern affects the approaches used, as Structured Light or Time-of-Flight devices (top left) contain a mostly dense depth with large missing areas, whereas LiDAR patterns (bottom left) resemble irregular grids.

the Microsoft Kinect [289], a consumer market Structured Light based depth camera, returns mostly dense low-resolution depth maps with noisy measurements and missing regions due to the regions being out-of-range, or being part of a non-reflective surface [307]. LiDAR depth maps are sparser, *e.g.*, less than 10% of valid depth per image in some cases [97]. Furthermore, LiDAR depth maps have a sampling pattern more akin to an irregular grid with each depth value is usually isolated from the rest. Compared to Structured Light sensors, the LiDAR sensors used in outdoor scenes, *e.g.*, autonomous driving, have a larger range and provide more accurate depth, although the measured depth still contains noise and artifacts. Due to the added noise in depth sensors, a depth completion method has also to correct the given input depth [333]. As for annotations, depth completion needs a denser (and less noisy) annotation compared to the input depth. One of the most used benchmarks, KITTI [331, 97], needed to accumulate multiple frames and apply a classical stereo approach, Semi-Global Matching, to clean the noise from dynamic objects [331], while still not providing a pixel-perfect ground-truth (see Section 5.4 and Figure 5.14 for ground-truth errors).

Deep Learning Methods. Even though some non-learning classical methods can use the RGB information for depth completion [87, 19, 307], learned methods provide a large increase in accuracy [258]. A standard design for a Deep Learning method is to follow a *encoder-decoder* approach, where we first encode the input signals (usually RGB and sparse depth) into a common latent space where fea-

ture fusion is possible, and then a decoder reconstructs an output depth map [333, 258, 175, 213, 353]. Several common trends can be found in depth completion architectures: use of Spatial Propagation Networks, confidence weighting, multiple streams and the use of several modalities. *Spatial Propagation Networks* variants [200, 56, 55, 244] refine depth predictions learning affinities from neighboring pixels and can be applied on top of depth completion models. *Multiple Streams* are helpful either to separate color and depth information before performing a late fusion [258, 130, 199], or to separate global and local paths [333]. These multiple streams usually have *confidence weighting* to account for the noise in the input depth data [258, 130, 333]. Different *input modalities* such as surface normals [258, 353] or 2D-3D representations to improve depth boundaries [52] can be leveraged to increase the accuracy of the model. It is quite common for depth completion architectures to combine several of these modules. For example, FusionNet [333] (used in Chapter 5) combined a *confidence map* to weight non-reliable input depth values, and a *multi-stream* approach with a global and local branch that are late-fused to obtain the final depth map.

Domain Adaptation Trends. Contrary to the complex pipelines needed to obtain real-data ground-truth [331], synthetic data presents pixel-perfect depth maps. However, the main challenge is *input adaptation* of the synthetic noiseless depth map to follow the distribution of a real depth sensor. Past LiDAR depth completion methods have used synthetic data without any adaptation module [11, 357, 258]. In the *synthetic LiDAR generation* step, [216] proposed a LiDAR simulator for detection and segmentation tasks using ray-casting and a learning process to drop points, while Wu *et al.* [344] analyzed the empirical point distribution to learn what synthetic points to drop. Other approaches [33, 231] tackle the synthetic data generation step by leveraging generative models to sample from them synthetic but real-like LiDAR scans (without generating a paired RGB image). For denser sensors, such as Time-of-Flight or Structured Light, a larger body of work has researched the adaptation synthetic to real depth. Some of the approaches relied on *adversarial image adaptation* [197, 281, 306, 112, 356], other works predicted the position of the depth holes in the synthetic data [9, 319], whereas some other methods modeled explicitly the sensors and noise sources [172, 255, 143, 303, 196]. The adapted synthetic depths can be then used in downstream tasks such as for example depth completion.

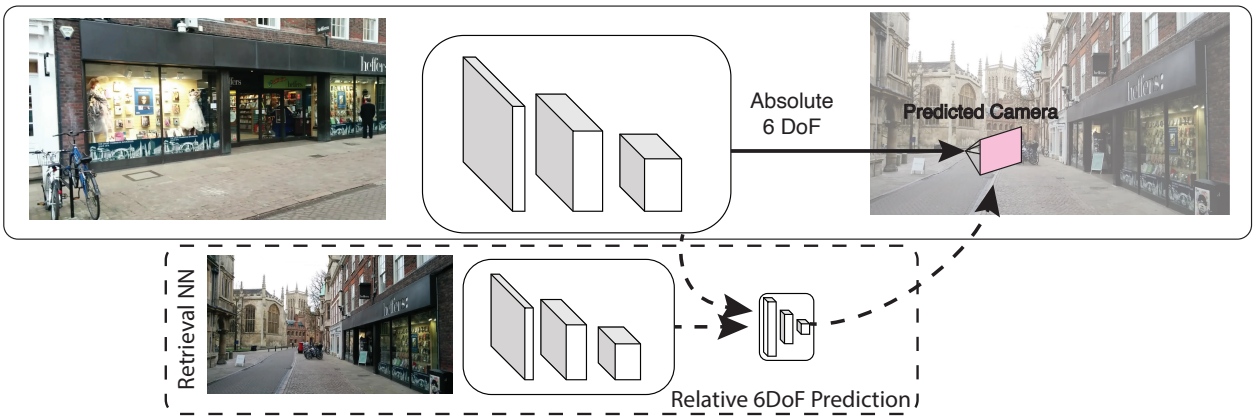


Figure 2.5: Camera pose estimation problem. We aim to estimate the camera pose where a single query image was taken from. We focus on regression-based methods, which can follow either an Absolute Pose Regression (APR) or a Relative Pose Regression (RPR) approach. In APR, the 6 Degrees-of-Freedom camera pose is directly estimated from a single image using a scene-specific coordinate frame (solid lines). RPR methods first retrieve a nearest neighbor with a known pose from the image database, which is combined with the query image to estimate a relative pose (dashed lines).

2.5 Camera Pose Estimation

Problem Setting. Camera pose estimation, also referred to as visual localization, aims to estimate the 6 Degrees of Freedom (DoF) (3DoF translation and 3DoF rotation) of a camera pose, which determines where an image is taken from in a given coordinate frame. Figure 2.5 shows the setup when using a regression approach for visual localization. Usually, in a pose estimation task, we assume that we have access to a collection of images with a known pose (database) and we want to estimate the unknown pose of a given query image. Query pose inference can be performed in several ways (*e.g.*, image retrieval, structure-based or camera pose regression), although in our work we focus on regression methods. Annotation for this task is performed via the use of a reference algorithm in the database images (and depth maps if available) to obtain a pseudo ground-truth pose. However, the algorithm chosen affects the quality of the poses and biases the evaluation results toward some methods [26]. On that note, synthetic data (*e.g.*, using a 3D-simulator) can provide a perfect ground-truth, but it has not been widely explored in the visual localization community [76]. We now briefly introduce camera pose regression methods, along with a process to compute depth maps from a collection of images, as both are relevant to the method we propose (see Chapter 6) for this task.

Depth Maps via Multi-View Stereo (MVS). Our camera pose estimation method in Chapter 6 lever-

ages synthetic data, which we generate projecting database images with known depth maps. In some cases, however, the database images do not include depth maps, hence we can use Multi-View Stereo to compute them. MVS uses a set of database images with known poses (computed, *e.g.*, using a Structure-from-Motion approach [302]) to estimate precise dense depth maps for the images in the database. A large number of MVS approaches, such as the method integrated in the widely used COLMAP [301] framework, use a variation of PatchMatch [18, 24]. Given two images, PatchMatch uses photometric consistency (*i.e.*, matches patches using the RGB values) to estimate a scale-less depth map that is mostly dense (except for some missing regions in, *e.g.*, highly reflective surfaces, textureless areas or dynamic objects).

Pose Regression directly estimates the camera pose from a CNN feedforward pass without minimal solvers [94] or RANSAC [88], which introduce a computational bottleneck. Regression approaches do not require an explicit 3D model either, however, their main drawback is the lower accuracy compared to structure-based methods [296]. As depicted in Figure 2.5, there are two types of regression methods. The first one is *Absolute Pose Regression (APR)*, which was proposed in PoseNet [154]. APR directly regresses scene-specific 6DoF poses from a single input image using a CNN [308, 122]. Hence, APR only requires a single feedforward pass for a given test image without any extra step, making them computationally efficient. Nevertheless, one of the main issues of APR methods is that performance-wise they are closer to image retrieval than to structure-based models [296]. To improve the performance of APR methods, past literature explored adding geometric constraints [30, 153], uncertainty measures [152] and using image sequences [338, 354]. The second type of regression method is *Relative Pose Regression (RPR)* [17, 75, 283], which computes the relative pose between the query and a database image. The database image used for RPR is ideally close in pose to the query image, and is obtained using image retrieval methods [260]. The predicted relative pose is combined with the known pose of the database image to obtain the final pose prediction. Contrary to APR, RPR methods are not necessarily scene-specific, however, currently RPR performance usually largely degrades when deploying them in an unseen scene [58]. RPR usually outperforms APR [296], as the distribution between training and testing sets are closer for relative poses [58].

Domain Adaptation Trends. The generation and use of *synthetic data* to increase the localization

performance is the most explored DA technique in visual localization. The use of synthetic views has been explored in place recognition [326], pose verification [321, 373] and pose estimation [232, 296, 339]. Mueller *et al.* [230] expanded the real database with synthetic data from a 3D model refined with an adversarial image adaptation method. [232, 339] generated views using predicted depths from a monocular model, which limited the synthetic poses to be close to existing ones. Other DA techniques have also been used, such as *adversarial image adaptation* in [257, 7], where query images were adapted to the weather and illumination conditions of the database images to improve feature matching and image retrieval. Another work [15] also tackled domain changes for feature matching, but instead of image adaptation, it used both a *consistency* approach using pairwise feature distances and a *divergence* CORAL-based method between the two distributions.

2.6 Datasets

We now present the datasets used for each of the presented tasks. Several of the presented datasets are driving-related as they are nowadays common benchmarks for outdoor depth and semantic tasks.

2.6.1 Object Detection

In our DA object detection work, which is presented in Chapter 3, we use the following datasets.

PASCAL VOC, which we use mainly as our source dataset in Chapter 3, contains images of real scenes that can be considered an unbiased representation of typical consumer images. There are 20 object classes annotated with bounding boxes and labels. As some defining statistics, 94.7% of images contain less than 7 instances, the average of instances per image is 2.42, and 98.7% of the images contain 3 or fewer unique class labels. Table 2.1 shows the distribution of instances per class.

Cross-domain Dataset was proposed in [140], and we use it as target dataset for most of our experiments in Chapter 3. Figure 2.6 (top) shows some image examples. There are three domains: Clipart1k, Comic2k and Watercolor2k. Clipart1k contains bounding boxes and labels for the 20 classes from PASCAL VOC, whereas Watercolor2k and Comic2k contain boxes and labels for only

Table 2.1: Number of available object instances in the dataset for each of the classes. VOC refers to the training and validation split from PASCAL VOC07 and VOC12. The rest refer to both train and test, which follow a 50/50 split.

Dataset	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	**Instances**	**Images**
VOC	1171	1064	1605	1140	1764	822	3267	1593	3152	847	824	2025	1072	1052	13256	1487	1070	814	925	1108	40058	16551
Clipart1k	73	36	265	129	121	21	202	50	340	46	115	54	79	17	1185	178	76	52	46	80	3165	1000
Watercolor2k	0	15	204	0	0	0	101	102	0	0	0	116	0	0	2483	0	0	0	0	0	3315	2000
Comic2k	0	87	270	0	0	0	107	233	0	0	0	192	0	0	5500	0	0	0	0	0	6389	2000

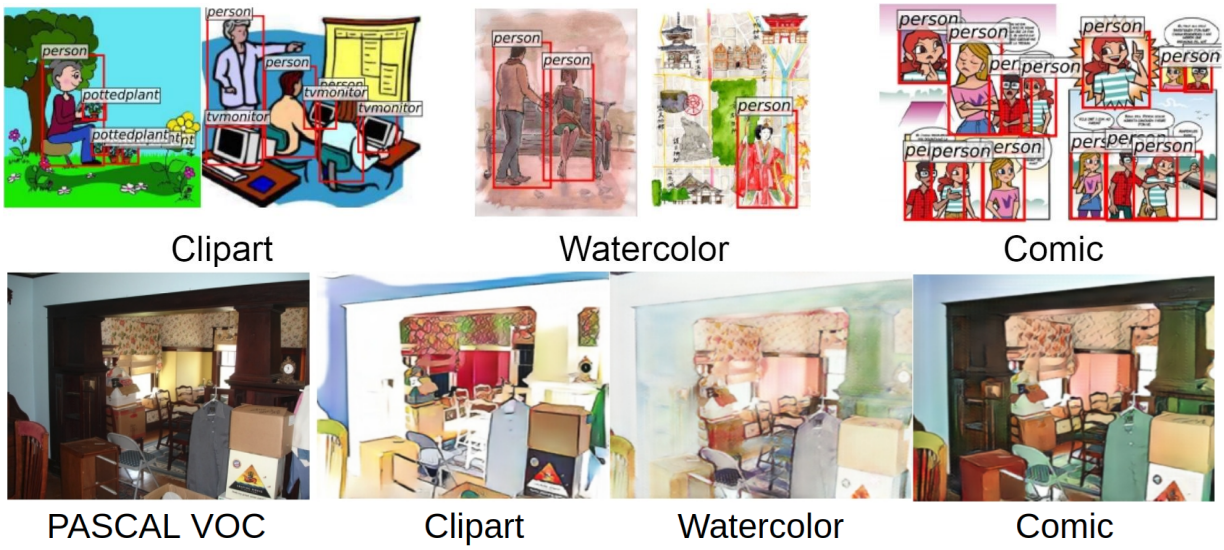


Figure 2.6: Top: Sample annotated images from the object detection Cross-domain Dataset from [140]. Bottom: Domain Transferred Images from [140].

6 of the PASCAL VOC classes. The datasets have a large class imbalance as Table 2.1 shows, which brings additional challenges. In Clipart1k, the category *person* accounts for more than one-third of the instances, whereas in Watercolor2k it accounts for around 75% of the instances, and 86% in the case of Comic2k. Figure 2.7 shows that each of the domains (clipart, watercolor and comic) contains instances with several different styles, with variations in the use of color, textures or patterns.

Domain Transferred Images (DTI) are images adapted from PASCAL VOC generated with Cycle-GAN [382] to match the style of either Clipart1k, Watercolor2k and Comic2k from the Cross-domain Dataset in [140]. Figure 2.6 (bottom) shows some examples of DTI.

Sim10k contains 10000 images collected from the video-game Grand Theft Auto V and annotated with instance-level boxes [149]. Sim10k is collected to mimic a driving dataset as the images are taken from a car view. Figure 2.8 shows an example of this dataset, along with the rest of the driving datasets we use for other tasks.



Figure 2.7: Samples from Clipart1k, Comic2k and Watercolor2k showing the diversity of styles inside a specific domain.



Figure 2.8: Examples of images from the driving datasets used in this thesis. Although they represent a similar scene, the datasets contain differences in illumination, weather and contrast among others.

Cityscapes is a driving dataset that was proposed in [64] for semantic understanding. It does not contain bounding boxes annotations, thus we follow [51] to retrieve ground-truth bounding boxes by finding the tightest rectangle around the instance mask, which are given in the dataset. Cityscapes contains 2975 images in the training set and 500 in the validation set. Following [51, 286] we use the validation set as the test set. Some samples are given in Figure 2.8.

Foggy Cityscapes was proposed in [288] and adds synthetic fog to Cityscapes using the depth information present in Cityscapes. The number of images is the same as Cityscapes, and it follows the same training and validation split. Some samples are given in Figure 2.8.

Summary. Current Domain Adaptation for object detection works mostly use two experimental settings: realistic to artistic transfer for common objects (*i.e.*, the classes present in PASCAL VOC) and autonomous driving-related datasets. A more general benchmark is needed for better evaluation

of Domain Adaptation for object detection pipelines. On that note, the development of a source synthetic dataset with common objects (similarly to the dataset developed in Peng *et al.*[251] for other tasks), coupled with the use of a diverse real dataset (*e.g.*, COCO or PASCAL VOC) as a target dataset, would provide a valuable setting for this task.

2.6.2 Depth Estimation and Depth Completion

The datasets used for our DA depth estimation and depth completion approaches are from an autonomous driving setting. For our depth estimation work in Chapter 4, our main experiment is the synthetic-to-real transfer of Virtual KITTI→KITTI, although Cityscapes→KITTI is also tested. As for our depth completion work in Chapter 5, we transfer from synthetic-to-real in CARLA→KITTI.

Virtual KITTI is a synthetic dataset that aims to mimic the scenes presented in KITTI. The dataset provides 50 videos (amounting to 21260 frames) with pixel-perfect metric-scale depth, flow, instance and semantic category maps, along with multi-object tracking ground-truth.

KITTI is a real dataset collected in Karlsruhe, Germany, using a car with multi-device setup on its roof [97]. The setup includes a stereo rig and a LiDAR device. The ground-truth used for the depth estimation pipeline is semi-dense as it comes from the LiDAR sensor, while the depth completion setting uses this LiDAR depth as input to the model. Hence, for depth completion, we need a denser and less noisy ground-truth, which is achieved by accumulating several LiDAR frames and cleaning them leveraging a Semi-Global Matching method [331].

Cityscapes, even though it is mainly used in semantic tasks (see Section 2.6.1), can also be used in a depth estimation context [64]. Cityscapes does not provide any LiDAR depth ground-truth, but it provides images from a stereo rig that are used to compute depth maps with a Semi-Global Matching technique. Semi-Global Matching returns a coarse scale-less depth ground-truth that is used sometimes to pretrain self-supervised monocular models [105]. In Chapter 4, we use Cityscapes along with the Semi-Global Matching depth maps as part of an extra evaluation setting .

CARLA is an autonomous driving simulator that can be used to extract synthetic data [78]. As it is a

simulator, it does not provide a standard split of training/test images, but allows us to collect multiple modalities from any user-defined set of cameras using expert driving agents, which can automatically drive the virtual car around the maps. For the depth completion experiments in Chapter 5 we use CARLA 0.84 to collect a set of multicamera RGB images along with the corresponding depth ground-truth under multiple weather conditions and maps.

Summary. There is a limited availability of large outdoor datasets with accurate metric-scale ground-truth. For that reason, KITTI [97] has become the main benchmark used for the evaluation of outdoor monocular depth estimation pipelines. Furthermore, synthetic simulators and datasets [92, 78] have also been developed for autonomous driving scenarios, which jointly with the real-domain KITTI dataset, provide the needed setup for a Domain Adaptation task. However, to properly assess the performance of monocular depth estimation models, we need to test the methods in datasets that represent a more diverse set of scenes and capture conditions. On that note, MiDAS [262] jointly merges multiple available datasets with depth labels to provide both a diverse training and test set. Nevertheless, the approach of merging the available datasets also presents its set of issues, such as the variable scale, quality and density of the depth annotations in the datasets. Concerning LiDAR depth completion settings, due to the complicated pipeline needed to obtain a semi-dense ground-truth, there is a larger reliance on testing the pipelines on KITTI. This lack of real-domain depth completion datasets is also coupled with the current scarcity of synthetic datasets that include both a realistic LiDAR depth and RGB images, as the current datasets either only simulate a limited range of noise sources [78, 139] or simplify the collision boxes when using a LiDAR ray-casting module [78]. The development of more diverse real and synthetic datasets for LiDAR depth completion would allow for a better evaluation of the depth completion models.

2.6.3 Visual Localization

For our camera pose estimation work (see Chapter 6) we use the standard datasets in regression settings, Cambridge Landmarks [154] and 7Scenes [103]. We show examples in Figure 2.9.

Cambridge Landmarks is an outdoor dataset that was proposed in [154]. It contains multiple scenes

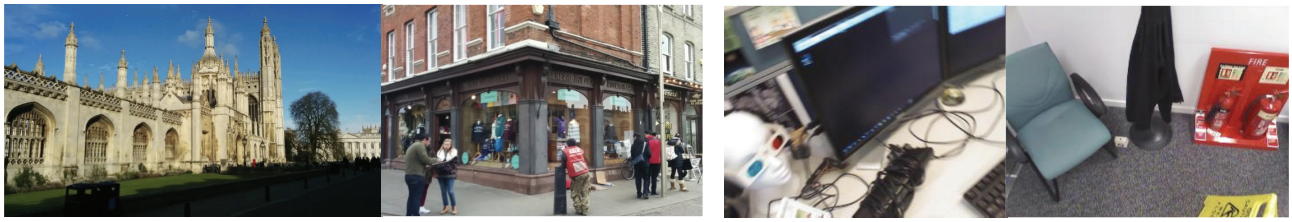


Figure 2.9: Examples of images from camera pose estimation datasets. *Cambridge Landmarks* [154] is the standard benchmark for pose regression in outdoor settings, whereas *7Scenes* [103] is the standard dataset for indoor pose regression settings.

covering a varying amount of surface area, where all of the scenes represent landmarks from Cambridge, UK. Each scene depicts a prominent landmark, with a wide variety of viewpoint and illumination changes, as well as dynamic objects. The spatial extent and number of training images vary per scene, for example, the smallest scene, *Shop Facade*, contains an extent of 35 m$\times$25 m with 231 training images whereas *Street* covers an extent of 500 m$\times$100 m with 3015 training images.

7Scenes is an indoor dataset with 7 different scenes, where the images come from Kinect [103] videos covering multiple trajectories. The Kinect videos include both depth and RGB information, however, the depth frames are not aligned with the RGB frames, hence in our pose estimation work we use the depth maps given in DSAC* [29], which were computed from a 3D model. Unlike Cambridge Landmarks, trajectories in 7Scenes are less structured and include more elevation changes. Furthermore, all of the scenes in 7Scenes are small and have a similar extent. The smaller scene, *Heads*, contains 1000 training images with an extent of 2 m$\times$1 m$\times$0.5 m, whereas the largest scene, *Red Kitchen*, contains 7000 training images and covers 4 m$\times$3 m$\times$1.5 m.

Summary. We test our visual localization approach on the standard camera pose regression benchmarks, *i.e.*, in *Cambridge Landmarks* and *7Scenes*. Although structure-based methods achieve high accuracy in the proposed benchmarks, these datasets are still challenging for camera pose regression works. However, both *Cambridge Landmarks* and *7Scenes* provide limited conditions variability, *e.g.*, the camera used and weather conditions are consistent for both dataset and query images. Further work in camera pose regression should tackle more realistic and complicated settings, such as the Aachen Dataset [295], which is widely used in the evaluation of structure-based methods and includes variable image capture conditions (*e.g.*, night-day images or variation of camera intrinsics).

A comparison of the currently used datasets in visual localization is given in [144].

2.6.4 Summary

We have introduced the main techniques used in Domain Adaptation works, along with the main tasks that we tackle in our proposed methods. Looking at the DA trends presented, the effectiveness of the techniques varies depending on the task, as each of the problems presents different adaptation challenges. Generally, in semantic-related tasks, feature adaptation, either via divergence-based methods or especially via adversarial methods, is investigated by more works (see Section 2.2) because it tends to improve the performance of a Domain Adaptation method. However, 3D-related tasks mostly approach the problem from an input adaptation perspective (see Section 2.3, Section 2.4, Section 2.5). In addition, 3D-related tasks are also more sensitive to undesirable distortions (*e.g.*, non-matchable areas for local features or artifacts for depth estimation) in the translated images both in the RGB image [311, 7], or in the adapted input depth [356], which provide a specific set of constraints when applying image translation.

The following chapters show that for different tasks we can adapt some of the main DA techniques, which are shown in Figure 2.1, to improve the results. As a common trend, the techniques applied to our approaches show that input data adaptation is a promising tool for all types of problems, especially if tuned to the specific task. On that note, we use three different forms of input adaptation depending on the task tackled. Furthermore, consistency (in the output prediction and between features) and pseudo-labelling are also applied successfully to our methods for semantic (*i.e.*, object detection) and depth-related tasks, which also points towards the wide applicability of those techniques. We also tackle the synthetic data generation step, and we show that if we tune the generation to our problem (*e.g.*, in terms of camera poses used or statistics of the generated data) the performance in the real domain can be further increased.

Chapter 3

Domain Adaptation for Object Detection via Style Consistency

This chapter addresses the object detection problem (see Section 2.2 for object detection background) from a Domain Adaptation viewpoint. We specifically introduce a method that is suited for a multi-style target dataset, *i.e.*, a dataset where the same class is represented in different visual styles. We adapt DA techniques, first image adaptation and feature consistency to make the detector robust to low-level differences, and then pseudo-labels to adapt the classifier to changes in the high-level features. The proposed method achieves state-of-the-art results in a multi-style benchmark.

3.1 Introduction

Typically, object detectors rely on a generic backbone (*e.g.*, VGG [308] or ResNet [122]) pretrained on ImageNet [71] for classification tasks, and then finetuned on datasets with similar images for both training and evaluation, such as MS COCO [195] and PASCAL VOC [83]. However, such models do not behave well when trained and tested in different domains as shown in [140, 51, 286].

Several approaches have been developed for Unsupervised Domain Adaptation to adapt the models, ranging from high-level feature adaptation via domain confusion [93, 330] to image adaptation using

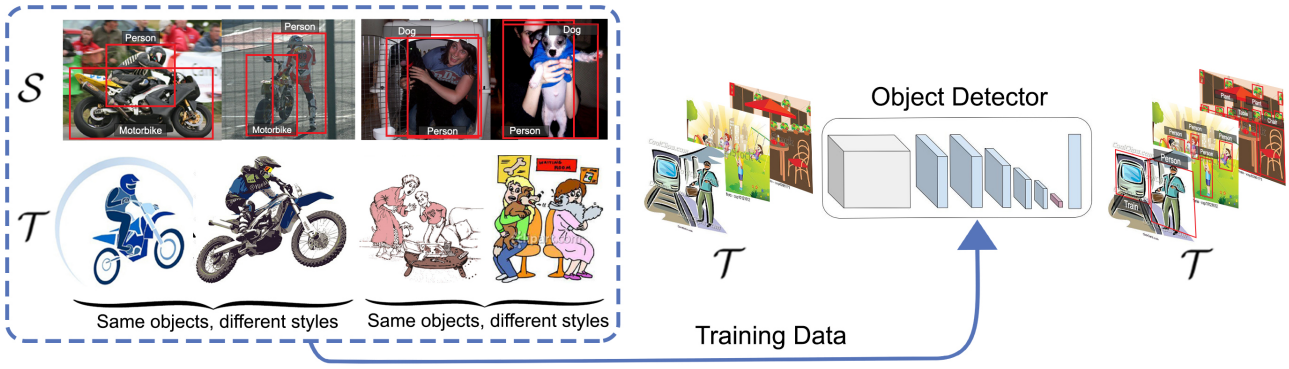


Figure 3.1: Overview of the unsupervised Domain Adaptation object detection task. We have a labeled source set $\mathcal{S}$ and an unlabeled target set $\mathcal{T}$ that shares classes with $\mathcal{S}$. We aim to use both sets to train a detector with good accuracy in $\mathcal{T}$.

a generative model [126, 329] (see Section 2.1.1 for explanation of Domain Adaptation trends, and Section 2.2 for Domain Adaptation for object detection related work). Image adaptation has been shown to be beneficial for closing the domain gap in several tasks, especially when the two domains are largely dissimilar in low-level statistics. The models usually employed for Domain Adaptation tasks are based on a CycleGAN [382] approach, thus they only provide one translated image per source image, and they cannot translate the source instance to a specific style in the target domain. However, some datasets may contain several styles as shown in Figure 3.1, *e.g.*, changes in texture, shapes, patterns, illumination or colors for a same class in the dataset. In those cases, having the capability of generating several translated source images individually to any of the target styles may help to close the domain gap.

Although past work [161] explored training multiple adversarial models to increase the generation diversity, [161] still cannot transfer to individually chosen styles. Conversely, style transfer methods are capable of adapting the input source image in an instance-level manner, making them suitable for a multi-style Domain Adaptation task. However, style transfer only tackles low-level changes in the domains, whereas the object appearance can also vary at a high level, *e.g.*, shape changes. Hence, for a good performance on the target domain, we also need to adapt the high-level features once the model is robust to the low-level changes, which can be done pseudo-labelling the target domain instances using the detector trained in the source domain.

In this chapter, we propose an approach for Domain Adaptation of object detectors that relies on these main contributions:

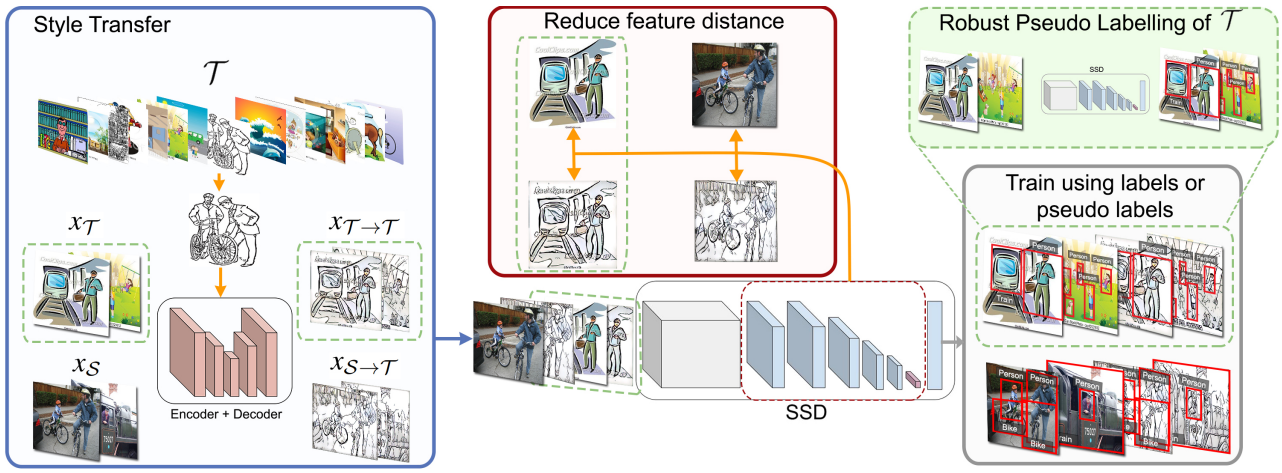


Figure 3.2: Overview of the proposed approach for Domain Adaptation of object detectors. We first make the detector robust to low-level variations by using a style transfer method for pixel-level adaptation of the source images, and then we force a high-level feature consistency between the style transferred image and the original image. As a final step, we pseudo-label $\mathcal{T}$, and finetune the model following the same pixel-adaptation and feature consistency method with both images from $\mathcal{S}$ and $\mathcal{T}$.

- We propose the use of an image adaptation method capable of transferring to individual styles in a setting where the target domain can have multiple visual styles. The results show that representation of multiple target domain styles leads to a better adaptation.

- High-level semantic features for object detection should be invariant to the style. We thus introduce feature consistency between the original and the style-transferred image to increase the robustness of the detector to low-level variations.

- We propose a pseudo-label strategy that is crafted for object detectors. Our pseudo-label strategy introduces reliable positive and negative examples from the target domain into training.

3.2 Method

In this section we present our approach, which is based on the SSD detector. We employ SSD as our object detector to be consistent with the prior work in Inoue *et al.* [140]. Our approach consists of two steps. In the first step, we train the object detector via style transfer and feature consistency to improve robustness to low-level variations. To adapt it to high-level changes, we provide the detector

with pseudo-labeled training examples from the target domain $\mathcal{T}$ to finetune the model. Figure 3.2 shows an overview of the method.

3.2.1 Single Shot Detector (SSD) Supervision

Our approach uses as base model the SSD detector [201]. SSD is a single stage approach with a set of anchor boxes from different feature maps. The number of boxes the detector outputs is fixed (8732 boxes for input images of 300×300), where the boxes vary in scale and aspect ratio. During training, each of the ground-truth boxes is matched to at least one of these 8732 boxes, and furthermore to any other box with an overlap higher than a set threshold. The training loss applied to these matched boxes for a given image x is

$$\mathcal{L}_{SSD}(x) = \frac{1}{N}\sum_{b}(\mathcal{L}_{conf}(b, s_c) + \alpha\mathcal{L}_{loc}(b, g)) \tag{3.1}$$

where the confidence term $\mathcal{L}_{conf}$ is the cross-entropy loss of a score s_c for ground-truth class label c averaged over all N matched boxes in image x. $\mathcal{L}_{loc}$ is a smooth l_1 loss between the predicted geometric parameters of box b and the ground-truth box g for positive examples. Hyperparameters in our experiments include $\alpha = 1$ and ratio 3:1 of negative to positive boxes in an image (more details in [201]). The loss $\mathcal{L}_{conf}$ for a single box b with predicted score s_c for correct class label c is:

$$\mathcal{L}_{conf}(b, s_c) = -\log(s_c) \tag{3.2}$$

where s_c is the normalized score after applying a softmax function.

3.2.2 Style Transfer

We now introduce the image translation approach used in our method. We adopt AdaIN [134] (see Section 2.1.1 for more style-transfer methods) to transfer the style of a target domain image $x_{\mathcal{T}}$ to a source domain image $x_{\mathcal{S}}$ via a feed-forward network resulting in $x_{\mathcal{S}\to\mathcal{T}}$. In AdaIN, a feature map is generated using a frozen VGG19 pretrained in ImageNet up to the layer `relu4_1` for target $f_v(x_{\mathcal{T}})$,

and source image $f_v(x_\mathcal{S})$. Next, $f_v(x_\mathcal{S})$ undergoes an affine transformation to match the mean and variance of $f_v(x_\mathcal{T})$ resulting in $f'_v(x_\mathcal{S})$. This affine transformation of the original source domain feature, $f'_v(x_\mathcal{S})$, is forwarded to a trainable decoder that generates $x_{\mathcal{S}\to\mathcal{T}}$. The decoder is trained using a style transfer loss, which is the sum of a content loss $\mathcal{L}_{Cont}$ and a style loss $\mathcal{L}_{Style}$:

$$\mathcal{L}_{ST} = \mathcal{L}_{Cont} + \mathcal{L}_{Style} \tag{3.3}$$

The content loss is $\mathcal{L}_{Cont} = \|f'_v(x_\mathcal{S}) - f_v(x_{\mathcal{S}\to\mathcal{T}})\|^2$, where $f_v(x_{\mathcal{S}\to\mathcal{T}})$ is the VGG19 `relu4_1` feature for $x_{\mathcal{S}\to\mathcal{T}}$. The style loss $\mathcal{L}_{Style}$ is the Euclidean distance between the means and the standard deviations of the feature maps from $x_\mathcal{T}$ and $x_{\mathcal{S}\to\mathcal{T}}$ in `relu1_1`, `relu2_1`, `relu3_1` and `relu4_1`.

3.2.3 Feature Consistency with Style Transfer

Past studies [99] have revealed that Deep Learning approaches are heavily reliant on low-level features for classification, making low-level adaptation crucial for Domain Adaptation. We thus aim to transfer the style of the target images to the source images to reduce the low-level domain gap. On that note, due to the multiple possible styles in the target domain, we leverage the AdaIN style transfer approach presented in Section 3.2.2 for low-level adaptation. During training, we sample at random a style image from the target domain $x_\mathcal{T}$ and use it to transfer each of the source images $x_\mathcal{S}$ into that target style, resulting in $x_{\mathcal{S}\to\mathcal{T}}$. Image $x_{\mathcal{S}\to\mathcal{T}}$ shares the ground-truth labels with $x_\mathcal{S}$, which we use to compute the loss $\mathcal{L}_{SSD}$ for both $x_{\mathcal{S}\to\mathcal{T}}$ and $x_\mathcal{S}$. Contrary to the low-level adaptation proposed in [140] with CycleGAN, which obtained one transferred image per $x_\mathcal{S}$ (without guarantee that all target domain styles are represented in the translation process), we are capable of generating several instances $x_{\mathcal{S}\to\mathcal{T}}$ out of the same $x_\mathcal{S}$ by sampling different style images $x_\mathcal{T}$ during training. Furthermore, we can also adapt to specific styles in the target domain, which is beneficial for multi-style datasets.

Enforcing Feature Consistency. Our motivation to train the detector with $x_{\mathcal{S}\to\mathcal{T}}$ instances is to force the network to recognize the objects under diverse low-level transformations. However, image adaptation, *i.e.*, introducing $x_{\mathcal{S}\to\mathcal{T}}$ examples during training, imposes no explicit constraints on the feature level. Compared to $x_\mathcal{S}$, the generated $x_{\mathcal{S}\to\mathcal{T}}$ contain low-level variations, *e.g.*, color, texture

or edges, but the location and shape of the objects is mostly preserved. Therefore, we assume that high-level features in the detector should remain invariant to the style change of the image. Following that assumption, we introduce a constraint to reduce the distance between the high-level features of $x_{\mathcal{S}\to\mathcal{T}}$ and $x_{\mathcal{S}}$, aiming to increase the robustness of high-level features to low-level changes. Hence, we introduce a feature consistency loss $\mathcal{L}_{Cons}$

$$\mathcal{L}_{Cons} = \sum_{k=1}^{K} ||f_k(x_{\mathcal{S}}) - f_k(x_{\mathcal{S}\to\mathcal{T}})||^2 \tag{3.4}$$

where $f_k(x)$ refers to each of the last $K = 6$ feature maps of SSD300, which are the maps used to detect the object instances. Figure 3.2 shows how $\mathcal{L}_{Cons}$ is applied to the SSD300 detector.

3.2.4 Robust Pseudo-Labelling

The approach presented in Section 3.2.3 makes the high-level features extracted by the detector robust to low-level changes, *i.e.*, the model extracts similar high-level features for both $x_{\mathcal{S}}$ and $x_{\mathcal{S}\to\mathcal{T}}$. However, the style transfer method does not incorporate any high-level semantic information in the image transfer process, meaning that there will still be a domain gap between the high-level features extracted by the detector from the images coming from the source domain $\mathcal{S}$ and the images coming from the target domain $\mathcal{T}$. We aim to adapt the classifiers to those high-level differences via a pseudo-labelling approach. A possible Domain Adaptation approach is to automatically label the target data using the source data trained model $h_{\mathcal{S}}(x_{\mathcal{T}})$ and then obtain an improved model $h_{\mathcal{S}+\mathcal{T}}$ by training with such pseudo-labeled examples. Similarly to the pseudo-labelling approach used in past works [62, 386], we use the confidence of $h_{\mathcal{S}}(x_{\mathcal{T}})$ to label $\mathcal{T}$. However, contrary to an image classification task, in an object detection problem we also need to carefully select negative examples.

Positive Examples Sampling has been done in [140] by taking top-1 object per class in a target image in a weakly supervised setting, which assumes knowledge of image-level labels, discarding other positive instances. However, applying the same approach without image-label knowledge, *i.e.*, pseudo-labelling the top-1 detected object for all classes, results in a drop of performance due to the large number of false positives. Furthermore, by only labelling the top-1 detection per class, we miss

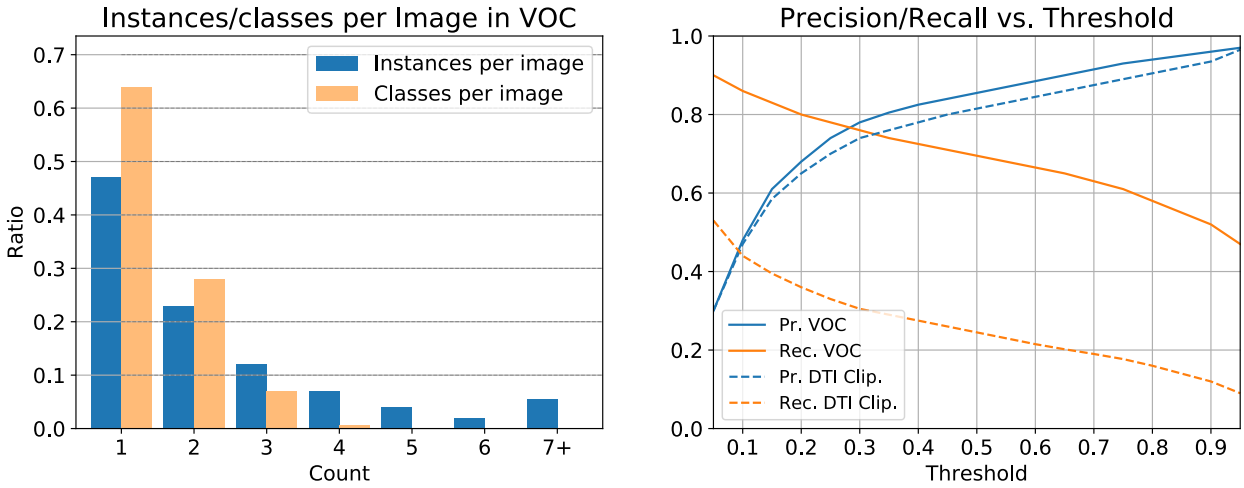


Figure 3.3: Left: Instances and unique classes per image in PASCAL VOC07/VOC12. Right: Precision and Recall curves for different recognition thresholds for $h(b)$ for the test set of PASCAL VOC07 and the training set of the *DTI* images with a model trained in PASCAL VOC07/VOC12.

on a large number of positive examples for those images containing multiple instances of the same class. On that note, Figure 3.3 shows that even though most of the images in PASCAL VOC [83] contain between 1-3 unique classes, a non-negligible number of images contain 7 or more instances, which points to a sizable ratio of images showing multiple instances of the same class. We argue that the actual value of the label score $s_c = h(b)$ for box b and class c output by the detector can be used to filter out the less confident detections. We therefore propose to accept a pseudo-labeled example if the probability of c being correct is $P(c|h(b)) \geq \epsilon_+$. The parameter ϵ_+ defines the precision of the sampling process, which we expect to remain high in different domains for a given high $h(b)$, while the recall may drop significantly. We test this assumption by evaluating a detector trained on the source domain for our main experiment, PASCAL VOC [83], and tested in an image-transferred version of PASCAL VOC, to which we refer as Domain Transferred Images (*DTI*), which presents strong low-level changes. *DTI* is built using a CycleGAN [382] to transfer PASCAL VOC images to a clipart, comic or watercolor style [140] (see Section 2.6.1 for an extended explanation of the datasets used). We report the results in Figure 3.3. The precision curve is high for $\mathcal{S}$ and DTI images, *e.g.*, the detector confidence of 0.8 corresponds to $P(c|(h(b) \geq 0.8)) \geq 0.8$ in both domains, while recall drops from 0.6 to 0.16. Experimentally, we directly set the threshold $h(b) \geq \hat{s}_c$, which is used to include the predictions as pseudo-labeled examples. The pseudo-labelling process is only run once after the detector is trained on $\mathcal{S}$ with our style-transfer and feature consistency modules.

Negative Examples Sampling is equally important. Hard-negative mining for SSD [201] returns, for a given image, the top-k boxes of ground-truth class *background* with the lowest confidence $h(b)$ of being detected as *background*. This approach gives good results when all ground-truth annotations are available. However, in the unsupervised setting with positive target examples pseudo-labeled by $P(c|h(b)) \geq \epsilon_+$ at high precision point, many false negative examples will have $P(c|h(b))$ just below ϵ_+. Therefore, including these rejected top detections as negatives may result in positive examples within the negative set. Hence, we set another threshold $h(b) \geq \hat{s}_-$ (*i.e.*, the box b is predicted as background with a higher confidence than $\hat{s}_-$) for selecting the negative samples, aiming to reduce the likelihood of introducing false negatives. We maintain the ratio of 3:1 negative to positive number of examples as in [201] and negative sampling is performed for each instance during training.

Combining Pseudo Labels and Style Transfer. A target domain image, $x_{\mathcal{T}}$, can also be transformed to follow the style of another target image $x_{\mathcal{T}}$ using the presented style transfer method, generating a new instance $x_{\mathcal{T}\to\mathcal{T}}$, with the same but unknown labels as $x_{\mathcal{T}}$. The pseudo-labels from $x_{\mathcal{T}}$ can thus be used with $x_{\mathcal{T}\to\mathcal{T}}$ to train the detector. Hence, the model is also trained on the target domain to detect high-level concepts under different low-level variations, which are generated from individual target domain styles. Furthermore, we can also apply our feature consistency module to the $x_{\mathcal{T}}$ and $x_{\mathcal{T}\to\mathcal{T}}$ examples. Our feature consistency approach is related to contrastive self-supervised approaches [276], however, instead of image perturbations (*e.g.*, color jitter) we transform a given target example to the style of another target example.

3.2.5 Combined Domain Adaptation Loss

Our final loss combines all the previously discussed terms:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{ST} + \lambda_2 \mathcal{L}_{Cons} + \lambda_3 \mathcal{L}_{\mathcal{S}} + \lambda_4 \mathcal{L}_{RPL} \tag{3.5}$$

In summary, our approach applies the style transfer loss $\mathcal{L}_{ST}$ (see Equation (3.3)) to obtain $x_{\mathcal{S}\to\mathcal{T}}$ examples. The consistency loss $\mathcal{L}_{Cons}$ (Equation (3.4)) reduces the distance between high-level SSD feature maps of $x_{\mathcal{S}}$ and $x_{\mathcal{S}\to\mathcal{T}}$. $\mathcal{L}_{\mathcal{S}}$ refers to detector loss $\mathcal{L}_{SSD}$ (Equation (3.1)) for both $x_{\mathcal{S}}$ and $x_{\mathcal{S}\to\mathcal{T}}$

Table 3.1: Object detection results in Average Precision (%) in the domain Clipart1k. We report results for different combinations of the loss proposed in Section 3.2, and we use $\mathcal{S}$ for source domain data, $\mathcal{T}$ for target, *DTI* for domain transferred images, *ST* for pixel level adaptation using style transfer, *C* for consistency loss, *PL* for the pseudo-labelling method from [140] and *RPL* for robust pseudo-labelling († means no style transfer applied to $\mathcal{T}$ examples). Methods marked with * use a Faster R-CNN with images of 600 pixels on the shorter side. Our approach brings significant improvements over baseline and other state-of-the-art methods from [140, 330, 51, 286].

	PASCAL VOC→ Clipart1k																			
Model	aero	bike	bird	boat	bttle	bus	car	cat	chair	cow	table	dog	horse	mbike	prsn	plnt	sheep	sofa	train	tv	**mAP**
Fully supervised																					
$\mathcal{T}$ [140]	50.5	60.3	40.1	55.9	34.8	79.7	61.9	13.5	56.2	76.1	57.7	36.8	63.5	92.3	76.2	49.8	40.2	28.1	60.3	74.4	55.4
Weakly supervised																					
DTI+PL [140]	35.7	61.9	26.2	45.9	29.9	74.0	48.7	2.8	53.0	72.7	50.2	19.3	40.9	83.3	62.4	42.4	22.8	38.5	49.3	59.5	46.0
Unsupervised																					
$\mathcal{S}$ [201]	19.8	49.5	20.1	23.0	11.3	38.6	34.2	2.5	39.1	21.6	27.3	10.8	32.5	54.1	45.3	31.2	19.0	19.5	19.1	17.9	26.8
ADDA [330]	20.1	50.2	20.5	23.6	11.4	40.5	34.9	2.3	39.7	22.3	27.1	10.4	31.7	53.6	46.6	32.1	18.0	21.1	23.6	18.3	27.4
DTI [140]	23.3	**60.1**	24.9	41.5	26.4	53.0	44.0	4.1	45.3	51.5	39.5	11.6	40.4	62.2	61.1	37.1	20.9	39.6	38.4	36.0	38.0
*DA-Faster** [51]	15.0	34.6	12.4	11.9	19.8	21.1	23.2	3.1	22.1	26.3	10.6	10.0	19.6	39.4	34.6	29.3	1.0	17.1	19.7	24.8	19.8
*Strong-Weak** [286]	26.2	48.5	**32.6**	33.7	**38.5**	54.3	37.1	**18.6**	34.8	**58.3**	17.0	12.5	33.8	65.5	61.6	**52.0**	9.3	24.9	**54.1**	49.1	38.1
Ours																					
ST	32.0	56.9	24.9	36.0	29.3	51.2	46.7	4.0	47.8	55.7	38.9	16.1	43.7	79.2	64.2	39.2	**21.8**	43.3	38.6	47.9	40.9
ST+C	35.0	57.3	24.7	41.9	28.0	56.8	49.1	9.9	49.3	55.6	44.0	**16.5**	42.3	83.1	65.0	42.8	17.7	43.9	42.0	52.6	42.9
ST+C+PL† *[140]*	34.8	51.9	23.2	43.8	29.8	58.6	48.2	6.9	46.1	56.2	40.6	20.0	37.6	84.2	60.3	40.2	19.9	37.8	34.6	48.8	41.2
ST+C+PL [140]	32.4	50.2	22.9	43.2	28.6	60.2	46.3	5.6	46.7	57.1	40.9	21.4	35.7	87.9	63.3	41.4	19.4	37.8	35.2	54.3	41.5
ST+C+RPL†	35.5	58.8	24.9	41.9	25.3	59.1	48.4	10.2	49.4	55.7	41.5	15.9	42.4	**84.3**	66.1	42.2	18.7	43.8	44.0	52.6	43.0
ST+C+RPL	**36.9**	55.1	26.4	**42.7**	23.6	**64.4**	**52.1**	10.1	**50.9**	57.2	**48.2**	16.2	**45.9**	83.7	**69.5**	41.5	21.6	**46.1**	48.3	**55.7**	**44.8**

images. Finally, $\mathcal{L}_{RPL}$ is also $\mathcal{L}_{SSD}$ but for $x_{\mathcal{T}}$ and $x_{\mathcal{T}\to\mathcal{T}}$ images based on our robust pseudo-labelling approach (Section 3.2.4). $\mathcal{L}_{ST}$ and $\mathcal{L}_{Cons}$ are also computed for $x_{\mathcal{T}}$ and $x_{\mathcal{T}\to\mathcal{T}}$ in the second step of training, *i.e.*, after pseudo-labelling. λ_1, λ_2, λ_3 and λ_4 are hyperparameters. We set the different λ_i values to 0.0 or 1.0 to activate or deactivate the different loss terms depending on the training step (see Section 3.3), we thus avoid using target domain data to further tune the hyperparameters. We investigate the impact of using other values for the different λ_i in Section 3.3.1.

3.3 Experiments

Architecture. We use the ChainerCV [239] implementation of SSD300, which employs a VGG16 as feature extractor and is pretrained on the *trainval* subset of VOC2007 and VOC2012. The input images are resized to 300x300 and during inference we discard regions with a confidence lower than 0.01. An intersection over union (IoU) threshold of 0.45 is used for Non-Maximum Suppression as

Table 3.2: Object detection results in Average Precision (%) in Watercolor2k and Comic2k.

	PASCAL VOC→ Watercolor2k							PASCAL VOC→ Comic2k						
Model	bike	bird	car	cat	dog	prsn	**mAP**	bike	bird	car	cat	dog	prsn	**mAP**
Fully sup.														
$\mathcal{T}$ [140]	76.0	60.0	52.7	41.0	43.8	77.3	58.4	55.9	26.8	40.4	42.3	43.0	70.1	46.4
Weakly sup.														
DTI+PL[140]	76.5	54.9	46.0	37.4	38.5	72.3	54.3	55.2	18.5	38.2	22.9	34.1	54.5	37.2
Unsup.														
$\mathcal{S}$ [201]	79.8	49.5	38.1	35.1	30.4	65.1	49.6	43.9	10.0	19.4	12.9	20.3	42.6	24.9
*DA-Faster**[51]	75.2	40.6	48.0	31.5	20.6	60.0	46.0	-	-	-	-	-	-	-
*Str.-weak**[286]	82.3	55.9	46.5	32.7	35.5	66.7	53.3	29.6	**20.1**	26.3	20.3	22.3	51.6	28.4
DTI[140]	**82.8**	47.0	40.2	34.6	35.3	62.5	50.4	43.6	13.6	30.2	16.0	26.9	48.3	29.8
Ours														
ST	78.2	55.7	47.0	41.2	33.0	67.0	53.7	49.5	16.4	37.6	20.8	30.9	57.1	35.4
ST+C	81.4	54.3	47.5	40.5	35.7	68.3	54.6	51.4	17.3	39.9	21.4	**31.9**	56.1	36.3
ST+C+RPL†	81.3	56.1	48.2	41.0	40.4	72.0	56.5	55.0	18.4	40.9	22.7	31.5	59.5	38.0
ST+C+RPL	79.9	**56.5**	**48.6**	**42.1**	**42.9**	**73.7**	**57.3**	**55.9**	19.7	**42.3**	**23.6**	31.5	**63.4**	**39.4**

in [201]. For the style transfer model we follow the implementation of [134], where the encoder is a VGG19 pretrained on ImageNet and frozen during training. The decoder mirrors the encoder and is pretrained using WikiArt [238] as style images and MS COCO [195] as content images.

Training Details. We finetune the detector using SGD with a learning rate of 10^{-5} and a momentum of 0.9 as in [140]. The training is done in two steps. In the first step, we set $\lambda_1 = 1.0$, $\lambda_3 = 1.0$ and $\lambda_4 = 0$, and either $\lambda_2 = 1.0$ or $\lambda_2 = 0.0$ depending on if feature consistency is enforced. Before the second step, we pseudo-label the target set with the updated object detector, and train again setting $\lambda_4 = 1.0$. We only pseudo-label $\mathcal{T}$ once, using a threshold of $\hat{s}_c = 0.7$ (see Section 3.2.4). For negative sampling, we set $\hat{s}_- = 0.9$. With the higher $\hat{s}_-$ compared to $\hat{s}_c$ we aim to avoid including many false negatives as, unlike positive sampling, negative sampling is continuously performed during training by selecting the hardest negative boxes satisfying $h(b) \geq \hat{s}_-$. However, the method is stable to lower values of $\hat{s}_-$ as discussed later in this section. In the first step, the batch only contains $\mathcal{S}$ instances and, in the second step, half of the examples come from $\mathcal{S}$ and half from $\mathcal{T}$. We use examples from $\mathcal{S}$ also in the second step to mitigate the effects of the incorrect pseudo-labels in $\mathcal{T}$. The batch size is 6 and we finetune the detector for 10000 iterations in the first step and for 5000 iterations in the second step. Due to the small number of test images, and few instances for some classes, the obtained mean Average Precision (mAP) has a high variance. We thus perform mAP evaluation in the test set every 100 iterations and we report the average mAP of the last 10 evaluations.

Comparison. Our main experiments are performed in the cross-domain dataset [140]. We report the per-class Average Precision and mAP using the ChainerCV evaluation code as [140]. We include the results from [140] for the weakly supervised method, *DTI+PL*, and the ablation study of only using the Domain Transferred Images (*DTI*), which were generated from PASCAL VOC using a CycleGAN. For Clipart1k and Watercolor2k, we include the results given in [286] for both their method, *Strong-Weak*, and *DA-Faster* [51]. Both [51] and [286] use Faster R-CNN with images of 600 pixels on the shorter side. For Comic2k, we run the official code of [286] with global and local adaptation, and context regularization. We include the *ADDA* [330] results in Clipart1k given in [140].

Evaluation Metrics. The main metric used for our experiments is the mean Average Precision (mAP). mAP is the mean of the per-class Average Precision (AP) for all of the classes in the dataset. To compute the AP, we need to compute the precision (P) and recall (R) defined as

$$P = \frac{TP}{TP + FP} = \frac{TP}{\#prediction} \qquad R = \frac{TP}{TP + FN} = \frac{TP}{\#groundtruth} \tag{3.6}$$

where TP is true positive, FP false positive and FN false negative. The test set predictions are ranked by confidence to form a precision-recall curve as the recall is monotonically increasing by definition. We compute the AP from the precision-recall curve using the PASCAL VOC 07 definition [83]. The VOC 07 definition computes the average of the precision obtained at 11 different recall levels (0.0, 0.1,..., 1.0). However, at each recall level r we take the maximum precision value in the precision-curve among the recall values higher than r.

3.3.1 Domain Adaptation for Object Detection Results

Table 3.1 shows the results obtained in Clipart1k and Table 3.2 in Watercolor2k and Comic2k. Compared to *DTI*, the performance increases in all three datasets with our style transfer method (*ST*), especially in Comic2k with an increase of 5.6%. The improved performance shows that generating multiple $x_{\mathcal{S}\to\mathcal{T}}$ from a single $x_{\mathcal{S}}$ is beneficial in datasets with multiple styles. Applying the consistency loss (*ST+C*) proves to be favourable in the three datasets, notably in Clipart1k with an increase of 2.0%. We explored the impact of style diversity in our style transfer module by restricting the target

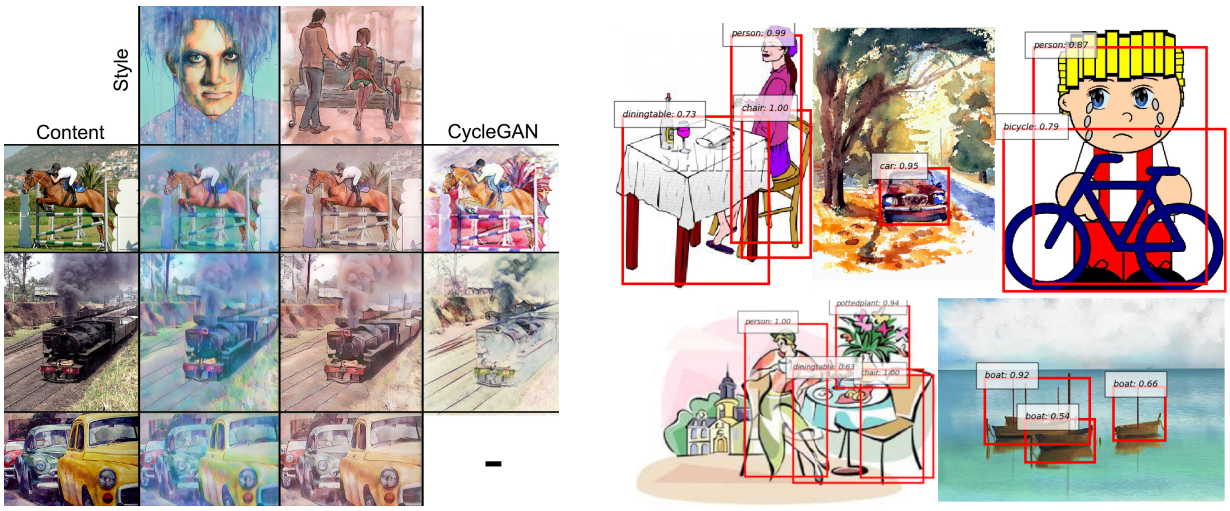


Figure 3.4: Qualitative results. Left: Examples of style transferred images and corresponding CycleGAN generated image in [140]. Right: Examples of object detections in the target domain.

style to a single random image from $\mathcal{T}$ in Clipart1k (with consistency loss) achieving 36.3%, which is 6.6% lower compared to using all of the styles. In the three datasets, using $x_{\mathcal{T}\to\mathcal{T}}$ instances along with their pseudo-labels (*ST+C+RPL*) leads to an improved performance compared to using only $x_{\mathcal{T}}$ instances (*ST+C+RPL*†). Even though the pseudo-labelling method proposed in [140] was developed for the case where image-level labels were available, we also tested its performance following their proposed approach for the unsupervised case, *i.e.*, taking the top-1 prediction for all classes and not using any threshold for the negative sampling. We show in Table 3.1 (*ST+C+PL* and *ST+C+RPL*†) that using the pseudo-labelling method from [140] results in lower performance compared to not using any pseudo-labels, which is due to three reasons. First, by only pseudo-labelling the top-1 prediction per class, high-confidence predictions that are not top-1 are discarded. Second, [140] uses a standard hard-negative sampling, selecting a high amount of false negatives during training. Third, when no image-level labels are given, [140] assumes that every class is present in the image, increasing the number of false positives. Using all the proposed modifications (*ST+C+RPL*) we obtain a higher mAP performance than the state-of-the-art: by 6.7% in Clipart1k, 4.0% in Watercolor2k and 9.6% in Comic2k. We also improve upon the weakly supervised method in [140] for Watercolor2k and Comic2k. Particularly, in Watercolor2k the mAP is only 1.1% lower than the fully supervised case.

Qualitative Results. In Figure 3.4 we show some $x_{\mathcal{S}\to\mathcal{T}}$ samples, along with the corresponding image generated by CycleGAN. The last row shows an example of $x_{\mathcal{T}\to\mathcal{T}}$. Figure 3.4 also shows examples of detections in $\mathcal{T}$. The model detects objects with similar shapes but different low-level statistics

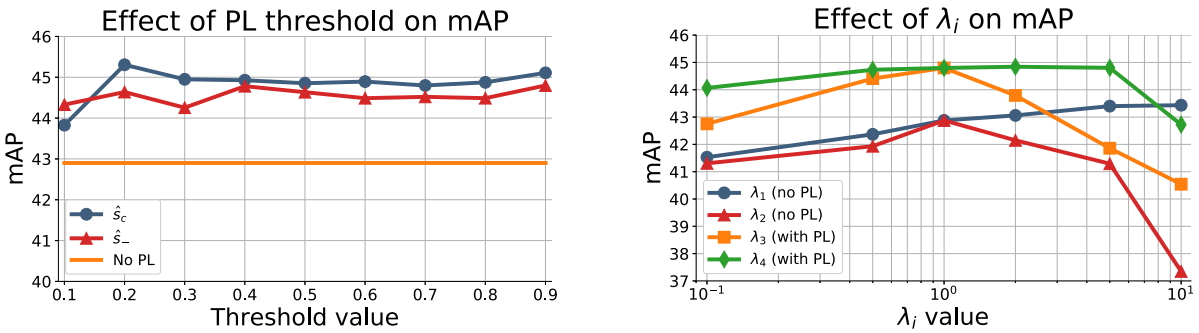


Figure 3.5: Left: Effect of the threshold used for pseudo-labelling in Clipart1k. Right: Effect of varying λ_i values in Clipart1k.

than the $\mathcal{S}$ domain, *e.g.*, the car in the upper row; objects with shape changes, *e.g.*, the person in the bottom-left image; and objects with a simplified shape, *e.g.*, the bicycle in the upper-right image.

Hyperparameter Sensitivity. Figure 3.5 shows the effect of varying $\hat{s}_c$ and $\hat{s}_-$ thresholds for pseudo-labelling (see Section 3.2.4 for definition of thresholds) in Clipart1k. The performance is stable for $\hat{s}_c \geq 0.2$ and low values of $\hat{s}_-$ give slightly lower performance. For the standard hard-negative mining case, *i.e.*, $\hat{s}_- = 0$, we obtained 44.0%, which is 0.8% lower compared to $\hat{s}_- = 0.9$, showing the importance of carefully selecting the negative samples. Figure 3.5 also shows the effect of varying λ_i in Clipart1k (see Section 3.2.5). For the first step of training, we vary the value of λ_1 and λ_2, which are the weights for $\mathcal{L}_{ST}$ and $\mathcal{L}_{Cons}$. We find that the performance increases when the style loss is given more weight. As for $\mathcal{L}_{Cons}$, $\lambda_2 = 1.0$ achieves the best result and $\lambda_2 = 10.0$ is the only value that drops the performance compared to $\lambda_2 = 0.0$ (see Table 3.1 results for *ST*, which uses $\lambda_2 = 0.0$). The reduction of mAP when setting λ_2 to larger values is expected as the model starts prioritizing consistency of features over task-performance, which could lead to a loss of semantic information. For the second step, we vary the weight of $\mathcal{L}_{\mathcal{S}}$ and $\mathcal{L}_{RPL}$ in the loss finding that the optimal setting is weighting them equally with $\lambda_3 = 1.0$ and $\lambda_4 = 1.0$. In this second step, we also see that increasing the value of λ_4 leads to a decrease of performance due to the model giving more weight to the pseudo-labels, which contain noise.

Driving Datasets. We follow the same protocol in [285] to test our method in two experiments: adaptation from Sim10k [149] (simulated data) to Cityscapes [64] (real data) and from Cityscapes to Foggy Cityscapes [288] (synthetic fog added to Cityscapes) as a second setting (see Section 2.6.1 for a detailed overview of the datasets). For these settings, we employed the SSD512 model, which uses

Table 3.3: Object detection results in Average Precision (%) in driving settings for two different settings, Sim10k→ Cityscapes and Cityscapes→ Foggy Cityscapes.

Sim10k→ Cityscapes		**Cityscapes→ Foggy Cityscapes**								
Model	**car**	bike	bus	car	mbike	prsn	rider	train	truck	**mAP**
Fully sup.										
$\mathcal{T}$	60.1	35.4	53.3	56.9	31.2	28.6	35.4	43.6	34.0	39.8
Unsup.										
$\mathcal{S}$	43.7	29.1	28.5	38.8	23.8	22.1	25.7	18.8	17.6	25.6
$\mathcal{S}$*[286]	34.6	26.5	22.3	34.3	15.3	24.1	33.1	3.0	4.1	20.3
*DA-Faster**[51]	38.9	31.0	25.0	40.5	22.1	35.3	20.2	20.0	27.1	27.6
*Str.-weak**[286]	42.3	**35.3**	**36.2**	**43.5**	**30.0**	**29.9**	**42.3**	**32.6**	**24.5**	**34.3**
Ours										
ST	43.7	29.7	32.6	40.8	26.1	24.1	30.3	20.9	23.4	28.5
ST+C	44.0	30.1	32.2	39.6	25.6	24.1	29.8	23.0	23.0	28.4
ST+RPL	44.1	29.6	32.3	42.1	26.9	24.4	30.2	24.3	22.4	29.0
ST+C+RPL†	42.1	28.4	31.3	39.0	23.4	22.0	24.7	21.1	22.0	26.5
ST+C+RPL	**44.2**	30.3	35.4	41.5	26.9	24.2	29.2	26.7	23.1	29.7

images of 512×512 pixels, due to small objects being present in the images. The batch size is reduced to 2 due to memory constraints, and for Cityscapes→ Foggy Cityscapes we doubled the number of iterations (20000 and 10000 iterations for first and second step). Table 3.3 shows the results. We also report the results given in [286] when using only source data for the Faster R-CNN method to show the difference with SSD512. Compared to *DA-Faster* [51] and *Str.-weak* [286], our method performs better in the Sim10k→ Cityscapes scenario and worse in Cityscapes→ Foggy Cityscapes. However, the different detector used (*i.e.*, Faster R-CNN or SSD) makes the results not directly comparable, as illustrated by the gap in performance of Faster R-CNN and SSD in the source domain in Table 3.3. Using our approach, the improvement is limited for Sim10k→ Cityscapes, whereas in Cityscapes→ Foggy Cityscapes we see an improvement of 4.1% over the source domain trained model. Style transfer adaptation is greatly beneficial for Cityscapes→ Foggy Cityscapes as Foggy Cityscapes only differs from Cityscapes in the low-level statistics, *i.e.*, added fog. Consistency only helps when using $\mathcal{T}$ instances, where it increases the performance by 0.7%. In both experiments, the best result is obtained when using all of the proposed blocks. Notably, using pseudo-labels without $x_{\mathcal{T}\to\mathcal{T}}$ instances drops the performance in both experiments, showing that forcing consistency in the detections between $x_{\mathcal{T}}$ and $x_{\mathcal{T}\to\mathcal{T}}$ instances is also beneficial in this setting.



Figure 3.6: Examples of images from driving datasets, and style transfer in those datasets. Left image is the content image, center image is the style image, and right image is the style-transferred image.

3.3.2 Limitations

In Figure 3.7 we include some examples of incorrect detections for the target artistic multi-style dataset. The class *person* is over-represented in the target datasets (see Section 2.6.1). This over-representation leads our detector to be biased to that class due to the large number of target-domain *person* pseudo-labels used during training. The effect of this bias is shown in the first three images in Figure 3.7, where our model incorrectly detects multiple instances as *person*. The model also presents issues merging partial detections for the same object, *e.g.*, the two *person* detections in the second image that correspond to the same instance, which may also be exacerbated by the use of noisy pseudo-labels. The artistic dataset contains anthropomorphic animals (*i.e.*, animals with a human form), where the detector incorrectly detects both an animal and *person* class (*e.g.*, joint *person*

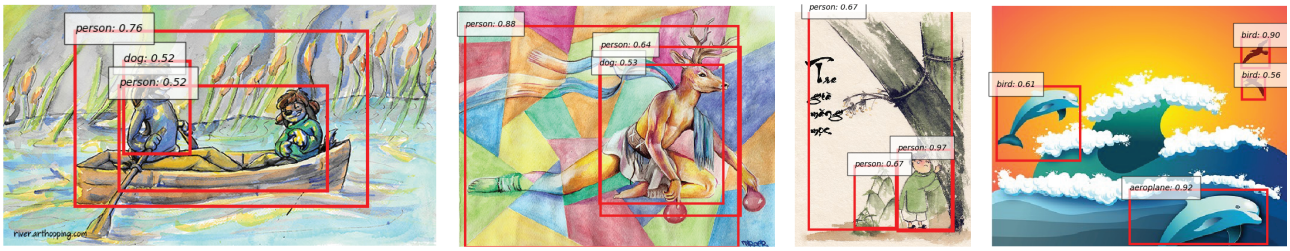


Figure 3.7: Examples of failed detections of our final model in the target domain for the adaptation from PASCAL VOC to multi-style artistic datasets.

and *dog* detection in the first two images) due to the strong shift with the real source domain. Including in the detector a stronger high-level adaptation would thus improve those detections. Lastly, the model uses some scene and background cues, and hence it detects incorrectly some objects that are in a context not present in the source dataset, *e.g.*, the dolphins (a class not present in our setting) jumping out of the water that are detected as a *bird* or an *aeroplane* in the last image. In the explored driving datasets, the improvements are limited for several reasons. Our method is suited for a multi-style target dataset, whereas in the driving settings the target datasets have a limited style variation, which reduces the benefits of image adaptation methods capable of individual style transfer, and also reduces the benefits of forcing feature consistency. Furthermore, our used style transfer method is not suited for realistic image generation, and the output images contain distortions as shown in Figure 3.6. Using methods adapted for realistic images such as [188, 360] may lead to an improved performance. Even though the investigated scenario of realistic to artistic-like images is of limited applicability, the proposed setting of having a multi-style target dataset is common in real-domain scenarios. Examples of multi-style real-domain datasets range from datasets merging different weather or illumination conditions [293, 13] to datasets containing multiple cameras with different color profile [347]. Coupling our approach with the mentioned photorealistic style-transfer methods [188, 360] could lead to an improved performance in such real-domain multi-style Domain Adaptation settings.

3.4 Conclusion

We presented an approach for Domain Adaptation of an object detector based on low-level adaptation via style transfer and feature consistency, and high-level adaptation via robust pseudo-labelling. We

demonstrated that each of these contributions brings an improvement with the highest performance obtained when all are combined. Our approach gives the highest mean Average Precision in the cross-domain dataset, where we proved the benefits of generating several translated source images to individual target styles. We also showed that combining style transfer with pseudo-labeled instances is key to improve the results. The method was also tested in driving datasets using the same parameters, showing an improvement over the source trained model. The comparison with other Domain Adaptation for object detection methods given in this chapter is against the approaches that were state-of-the-art when our original method was published [208]. The current state-of-the-art method is the Unbiased Mean-Teacher (UMT) [72] approach, which achieves a mAP of 44.1 in PASCAL VOC→ Clipart1k, 58.1 in PASCAL VOC→ Watercolor2k, 41.7 in Cityscapes→ Foggy Cityscapes, and 43.1 in Sim10k→ Cityscapes. UMT uses Faster R-CNN with a larger image resolution (shorter size is 600 pixels), and leverages a CycleGAN translation approach coupled with a teacher-student framework. Overall, current methods have mostly settled on using Faster RCNN as the de facto object detector used in Domain Adaptation [72, 45, 384], while image translation continues to be one of the main techniques used to obtain good performance [72, 45]. On that note, the competitiveness of our method to current state-of-the-art in the real-to-artistic setting (*i.e.*, our method achieves similar performance to UMT in that setting) shows the benefits of using a multi-style aware image translation approach for that task.

Chapter 4

DESC: Domain Adaptation for Depth Estimation via Semantic Consistency

With the methods proposed here, we aim to improve the domain transfer in a depth estimation task by using semantic information to guide the output depth map (see Section 2.3 for depth estimation background). We build our approach upon the natural relationship existing between depth and semantics, *i.e.*, perceptual principles, which are used by humans when inferring depth [74]. Furthermore, input data adaptation is combined with our semantic consistency module by leveraging adversarial image adaptation methods available in the literature. The proposed approach is evaluated on standard Domain Adaptation benchmarks for monocular depth estimation and shows consistent improvement upon the state-of-the-art.

4.1 Introduction

State-of-the-art depth estimation methods are capable of inferring an accurate depth map from a monocular image by relying on Deep Learning methods, which require a large amount of data with annotations [91, 170]. Annotations in the form of precise depth measurements are typically provided by special tools such as a LiDAR sensor [97] or structured light devices [307]. Obtaining depth annotations is thus costly and time-consuming. Much research has focused on developing

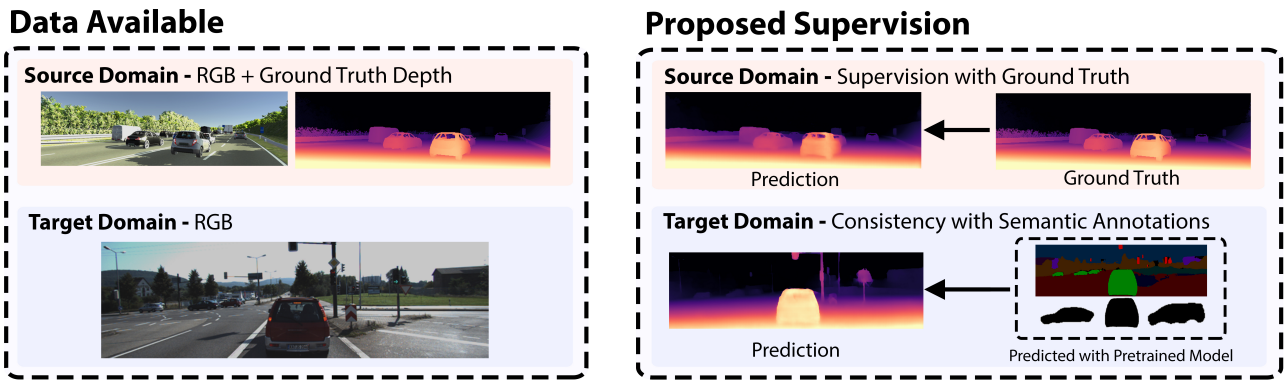


Figure 4.1: Overview of the data available and proposed supervision. The source domain $\mathcal{S}$ contains both RGB and ground truth depth data, and the target domain $\mathcal{T}$ contains RGB data only. We train a depth estimation model to achieve high performance in $\mathcal{T}$ by leveraging semantic annotations to introduce semantic consistency in $\mathcal{T}$. The semantic annotations are obtained using a panoptic segmentation model trained with external data.

methods not relying on directly acquired depth annotations by leveraging stereo [105, 95] or video sequences [104, 40, 359] for self-supervision. These research directions have shown promise, but a stereo pair or video sequence may not always be available in existing datasets. Synthetic data can be used to obtain a large amount of accurate ground truth depth in a fast manner, but the domain gap makes training with synthetic data challenging. On that matter, there has been Domain Adaptation works in a depth estimation context [233, 379] (see Section 2.3 for DA for depth estimation literature review) that helped to transfer the models trained on an annotated source dataset $\mathcal{S}$ to a target dataset $\mathcal{T}$, reducing the burden of training a model for a new environment or camera.

Research results have shown that the domain gap for semantic segmentation and instance detection can be reduced by introducing depth information during training [197, 337, 50, 282]. Leveraging semantic information to reduce the domain gap in depth estimation settings has been less studied and mainly in multi-task scenarios [12, 168]. Existing datasets with semantic annotations are large and varied in the scenes as well as cameras used, hence models trained on these diverse semantic datasets are capable of generalizing to different settings [171]. Several works [192, 40] have shown that using pretrained models to obtain semantic annotations can also bring improvements in the depth estimation task. Motivated by these findings, we exploit readily-available panoptic segmentation models as guidance to bridge the gap between two different domains and to improve monocular depth estimation. Figure 4.1 shows an overview of the task.

Domain Adaptation approaches benefit from pseudo-labelling [44, 285] and consistency of predictions in the source and target domains [374, 53]. Therefore, we propose an approach that leverages semantic annotations to introduce consistency and pseudo-labels in our depth estimation setting, leading to state-of-the-art results in the VirtualKITTI [92] to KITTI [97] setting among those works not using self-supervision. Our main contributions are:

- We introduce a novel consistency constraint with predictions from a second model trained on high-level semantics and low-level edge maps.
- We propose an approach to form depth pseudo-labels in the target domain by using object size priors, which are learned in an instance-based manner in the annotated source domain.

4.2 Related Work

In Section 2.3 we introduced the literature for Domain Adaptation for depth estimation. Now we aim to introduce the work that, similarly to our method, augments the depth estimation problem with semantic data.

4.2.1 Depth and Semantic Information

Depth and semantic information have been utilized simultaneously to improve depth estimation. The two predominant trends involve either using a multi-task approach to improve the depth predictor features, or using the semantic masks to regularize (*e.g.*, smoothness, edge alignment) and/or filter (*e.g.*, dynamic objects for self-supervision) the depth maps.

Multi-task learning of depth and semantic tasks has been proposed in multiple works. Mousavian *et al.* [228] trained a single network for both semantic and depth prediction in a multi-task manner by using a shared backbone and task-specific layers. On that note, Chen *et al.* [46] trained a network capable of selecting between depth or semantic segmentation output by only changing an intermediate task layer. Several works [146, 61, 184, 371] proposed novel units to share information between

the two tasks, which improved the depth features. In [12] the availability of temporal information was assumed in training and test time to fuse multiple frames for depth and semantic segmentation prediction. Guizilini *et al.* [113] used a pretrained semantic segmentation network to guide the feature maps of the depth network using pixel-adaptive convolutions.

Regularization of Depth with semantic information has also been used in past works to improve depth prediction quality. MegaDepth [192], a diverse Structure-from-Motion and Multi-View Stereo depth dataset collected from the internet, used semantic information to filter spurious depth values and to define ordinal labels that were used as extra guidance for the network. Some works used pre-computed object instances masks by filtering the dynamic objects from the photometric loss in a video self-supervision setting [40, 222, 165]. Related to our work, Struct2Depth [40], apart from filtering dynamic objects from the photometric loss, also used predicted instance masks to impose an object size-depth constraint by learning a single object height for all the *car* instances. Klingner *et al.* [165] trained a semantic segmentation branch used to detect dynamic objects, filtering out those that are moving while learning from those that are static (*e.g.*, parked cars). Zhu *et al.* [383] used semantic maps to regularize the depth edge of object instances using a morphing operator and a consistency loss, which aimed to tackle the bleeding artifacts in a stereo supervised method. Following this depth edge regularization approach, in [280] panoptic maps were employed to force a depth discontinuity in the instance edges, and also for stereo consistency, which alleviates some issues with photometric consistency (*e.g.*, non-Lambertian surfaces).

4.3 Method

In this section, we introduce our Domain Adaptation for Depth Estimation via Semantic Consistency (DESC) approach. An overview is presented in Figure 4.2. During inference we only apply our depth estimation network G_D to our target images, except when using other image translation strategies in Section 4.4.2. Semantic annotations are predicted for our source and target datasets using a panoptic segmentation model [164] trained with external data, where the semantic annotations provide for each image the detected instances and a semantic segmentation map.

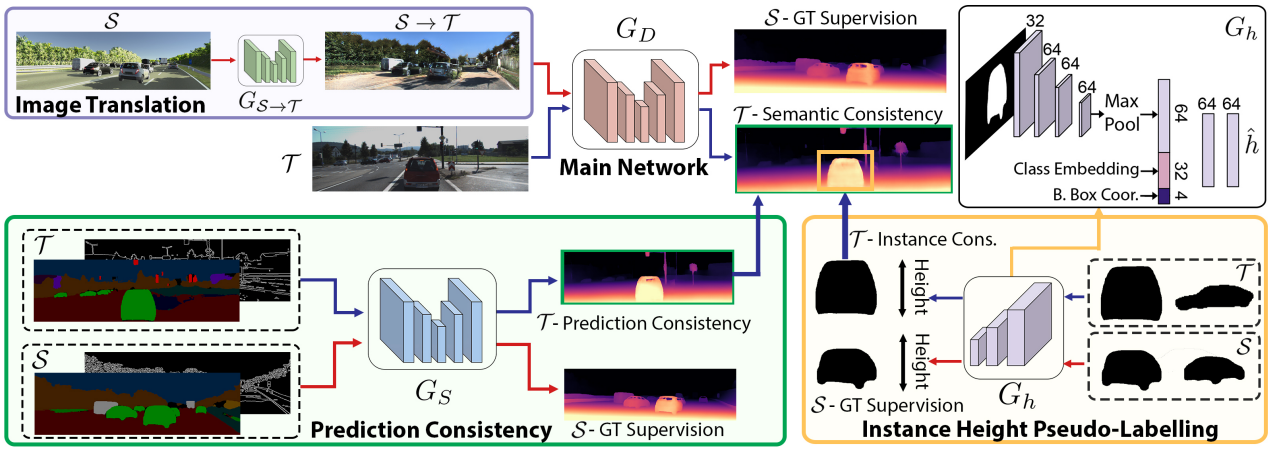


Figure 4.2: Overview of the approach. We train a depth estimation network G_D with both target $\mathcal{T}$ and source $\mathcal{S}$ images. Source images are adapted to the style of the target images. For $\mathcal{S}$, we use ground truth supervision, while we enforce consistency with semantic information in $\mathcal{T}$. The consistency is enforced with (1) predictions from a second network G_S trained with edges and semantic maps as input, and (2) depth pseudo-labels formed using an instance height $\hat{h}$ predicted by G_h. Both G_S and G_h are trained using ground truth data from $\mathcal{S}$. The architecture of G_h is given in the top right. We use ReLU between the layers of G_h.

4.3.1 Pseudo-Labelling using Instance Height

The height of the detected object instances can provide a strong cue for distance estimation, hence we aim to use the detected instances to provide a guidance in the target domain by generating pseudo-labels from the predicted height. To do so, we leverage the work in Struct2Depth [40], which used the instance height to deal with moving objects in video self-supervision and it retrieved an approximate distance to the objects by solving

$$\hat{D} \approx \frac{f \cdot h}{H} \tag{4.1}$$

where $\hat{D}$ is an approximate distance to the object, f is the focal length in pixels, H is the predicted instance size in pixels and h is the physical height of the object. It is assumed that the entire object instance is placed at a distance $\hat{D}$, that f is known, and that the real object size h is unknown. In Struct2Depth [40], the object size was set as a shared learnable parameter $\hat{h}$ for the class *car*, *i.e.*, all of the detected instances of class *car* were assumed to have the same height. We argue that predicting a $\hat{h}$ per object instance rather than class can provide a better height estimate, as it can take into account both intra-class variations and occlusions in the detected instances. Furthermore, instead of

learning $\hat{h}$ in an unsupervised manner as in Struct2Depth [40], we can improve the estimation using source domain data. Therefore, we use a simple network G_h (architecture presented in Figure 4.2), to predict a $\hat{h}_i$ for an instance i from the dimensions of its bounding box, the detected binary mask and the predicted class. We train G_h using labels in the source data by retrieving $h_{GT,i}$, which is the ground truth physical object size for instance i. To retrieve $h_{GT,i}$ we use $h_{GT,i} = \frac{H_i \cdot \hat{D}_{\mathcal{S},i}}{f_{\mathcal{S}}}$, where the instance depth $\hat{D}_{\mathcal{S},i}$ is obtained directly from the depth ground truth. To obtain $\hat{D}_{\mathcal{S},i}$ we use $\hat{D}_i = median(M_{\mathcal{S},i} \odot y_{\mathcal{S}})$, where $M_{\mathcal{S},i}$ is the binary segmentation instance mask for a source domain detected instance i, $\odot$ refers to the Hadamard product, $y_{\mathcal{S}}$ is the ground truth depth, and the median operation is performed only for non-zero values. Thus, G_h is trained on the source domain with $\mathcal{L}_{I,\mathcal{S}} = \frac{1}{n_I} \sum_i |\hat{h}_{\mathcal{S},i} - h_{GT,i}|$, where n_I is the number of detected instances. In the target domain, G_h is used to predict a height $\hat{h}_{\mathcal{T},i}$ for a detected instance i, and then $\hat{h}_{\mathcal{T},i}$ is used to retrieve a depth pseudo-label $\hat{D}_{\mathcal{T},i}$ computed using Equation (4.1). We use the depth pseudo-labels $\hat{D}_{\mathcal{T},i}$ to supervise G_D in the target domain using a sum of pixel-wise L_1 losses over all detected instances i. For this work, we use an L_1 loss for all depth-related losses to be consistent with the work we build upon, T^{2}Net [379], and other Domain Adaptation for depth estimation works [374, 49, 54]. Furthermore, L_1 losses tend to work well in depth estimation tasks [85]. Our loss is then

$$\mathcal{L}_{I,\mathcal{T}} = \frac{\phi}{p_I} \sum_i \| (\frac{\hat{D}_{\mathcal{T},i}}{\phi} - G_D(x_{\mathcal{T}})) \odot M_{\mathcal{T},i} \|_1 \tag{4.2}$$

where p_I is the sum of non-zero pixels for all the binary segmentation masks $M_{\mathcal{T},i}$, $x_{\mathcal{T}}$ is an image from $\mathcal{T}$ and ϕ is a learnable scalar. The scalar ϕ is used to correct any scale mismatch in the predictions of $G_D(x_{\mathcal{T}})$ due to camera differences between $\mathcal{S}$ and $\mathcal{T}$ [123]. When computing $\hat{D}_{\mathcal{T},i}$ we use the focal length $f_{\mathcal{T}}$ of the target domain camera, although as we will show in Section 4.4, ϕ automatically scales the values to the correct range even for unknown $f_{\mathcal{T}}$. As we use a panoptic segmentation model trained with external data to extract semantic annotations, some of the classes detected may be present in $\mathcal{T}$ but not in $\mathcal{S}$, *e.g.*, *person* in Virtual KITTI→KITTI. For those classes, G_h can also learn an instance-based height prior in an unsupervised manner via consistency with G_D in $\mathcal{L}_{I,\mathcal{T}}$.

4.3.2 Consistency of Predictions using Semantic Information

Many works [276, 89] have shown that constraining the learning process by requiring consistency in a Domain Adaptation setting reduces the performance gap (see also Section 2.1.1). Similar observations have been made in self-supervised learning [47], where a contrastive loss is used between different views of the same scene obtained via data augmentation. Following these findings, we enforce consistency between the predictions generated by our main depth estimation network, G_D, and a secondary network, G_S. Instead of using data augmentation techniques to generate another view to input to G_S, we aim to use low domain-gap modalities to increase the generalization ability of the network. Hence, input data x_{Sem} is formed by two channels that have a low domain gap: a semantic segmentation map and an edge map.

Semantic Structure. A semantic segmentation map provides information on the high-level structure of the scene, and this high-level structure helps to predict the depth structure. Similarly to the observation made by [380], we notice that datasets are more similar in their high-level structures or presented semantic scenes compared to their RGB similarity, where there are differences due to the quality of textures, illumination or models in synthetic data, among others. The semantic segmentation map is introduced in the form of an integer corresponding to the semantic class label, as we experimentally found it to yield better performance than one-hot encoding.

Edge Map. Deep Learning networks tend to use texture cues [99] for predictions. We use an edge map to reduce the impact of the texture differences between domains, and to provide a different data modality to the network. Furthermore, edge maps include information about the shapes of objects that is valuable in depth related tasks [129, 136]. Edges also give complementary information to the semantic maps and, compared to RGB images, present less variation and need less adaptation in domains with semantically similar scenes.

Consistency. As both networks G_D and G_S receive different input modalities, forcing consistency between them for the predictions of the target domain can significantly increase the target-domain performance of both models. We propose to supervise G_S with source domain depth ground truth y_S by using a pixel-wise L_1 loss, $\mathcal{L}_{Con,S}$, and then force consistency of predictions in the target domain

via $\mathcal{L}_{Con,\mathcal{T}}$. Assuming N is the total number of pixels,

$$\mathcal{L}_{Con,\mathcal{S}} = \frac{1}{N}\|G_S(x_{Sem,\mathcal{S}}) - y_{\mathcal{S}}\|_1$$
$$\mathcal{L}_{Con,\mathcal{T}} = \frac{1}{N}\|G_D(x_{\mathcal{T}}) - G_S(x_{Sem,\mathcal{T}})\|_1 \tag{4.3}$$

4.3.3 Training Loss

We now present the modules used in DESC in addition to our semantic consistency losses.

Depth Estimation Loss. Our model G_D outputs a multiscale prediction that is supervised using source domain ground truth with $\mathcal{L}_D$, which is a pixel-wise L_1 loss [379, 374]. The ground truth is resized to match the resolution of each of the maps output by G_D.

Image Translation has been demonstrated to effectively reduce the domain gap [374, 379]. We adopt the approach from T^{2}Net [379], where a network $G_{\mathcal{S}\to\mathcal{T}}$ translates the source image to the target domain without cycle consistency (see Section 2.1.1 for cycle consistency explanation). T^{2}Net [379] uses a least-squares adversarial term $\mathcal{L}_{GAN}$ [218] to produce examples $x_{\mathcal{S}\to\mathcal{T}}$ having a similar distribution to $x_{\mathcal{T}}$, and leverages the constraint imposed by $\mathcal{L}_D$ to ensure $x_{\mathcal{S}\to\mathcal{T}}$ is geometrically consistent with $x_{\mathcal{S}}$. The method also uses a L_1 identity loss $\mathcal{L}_{IDT} = \frac{1}{N}\|G_{\mathcal{S}\to\mathcal{T}}(x_{\mathcal{T}}) - x_{\mathcal{T}}\|_1$ to force $G_{\mathcal{S}\to\mathcal{T}}(x_{\mathcal{T}}) \approx x_{\mathcal{T}}$, *i.e.*, $\mathcal{L}_{IDT}$ forces $G_{\mathcal{S}\to\mathcal{T}}$ to behave as an identity mapping for $x_{\mathcal{T}}$. We also present results in Section 4.4.2 when using newer image translation techniques such as SharinGAN [256] and S2R-DepthNet [49], which yield improved results compared to using the T^{2}Net-approach.

Smoothing. We use for the target data the smoothing term $\mathcal{L}_{Sm}$ introduced in Monodepth [105], and successfully used in Domain Adaptation [379, 374] methods for depth estimation.

Overall Loss. Our final model is trained using the following loss

$$\mathcal{L} = \lambda_{\mathcal{S}}(\mathcal{L}_{\mathcal{D}} + \mathcal{L}_{Con,\mathcal{S}} + \mathcal{L}_{I,\mathcal{S}}) + \lambda_{\mathcal{T}}(\mathcal{L}_{Con,\mathcal{T}} + \mathcal{L}_{I,\mathcal{T}}) + \lambda_{Sm}\mathcal{L}_{Sm} + \lambda_{IDT}\mathcal{L}_{IDT} + \lambda_{GAN}\mathcal{L}_{GAN} \tag{4.4}$$

where $\lambda_{\mathcal{S}}, \lambda_{\mathcal{T}}, \lambda_{Sm}, \lambda_{IDT}, \lambda_{GAN}$ are hyperparameters to balance the different terms. The values of $\lambda_{\mathcal{S}}, \lambda_{Sm}, \lambda_{IDT}, \lambda_{GAN}$ are selected based on past work, whereas we set $\lambda_{\mathcal{T}} = 1.0$ and also investigate the impact of varying its value in Section 4.4.

Table 4.1: Results for Virtual KITTI→KITTI in KITTI [97] Eigen [82] split. For a fair comparison, we use the official pretrained models given by *T²Net* and *S2R-DepthNet* to recompute the results using median scaling. *S³Net* is the best reported result in [54] not using any semantic maps at test time, whereas *S³Net (Test Semantic)* uses semantic maps both at train and test time. S2R-DepthNet [49] is a domain generalization method that does not use any KITTI data during training.

	Lower is better				Higher is better		
Method	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Cap 80m							
AdaDepth [233]	0.214	1.932	7.157	0.295	0.665	0.882	0.950
T²Net [379]	0.173	1.396	6.041	0.251	0.757	0.916	0.966
S2R-DepthNet [49]	0.162	1.339	5.684	**0.232**	0.786	**0.934**	**0.974**
DESC	**0.156**	**1.067**	**5.628**	0.237	**0.787**	0.924	0.970
Cap 50m							
AdaDepth [233]	0.203	1.734	6.251	0.284	0.687	0.899	0.958
T²Net [379]	0.165	1.034	4.501	0.235	0.772	0.927	0.972
S2R-DepthNet [49]	0.155	0.997	4.327	0.220	0.799	**0.941**	**0.978**
S³Net [54]	0.154	0.993	4.449	0.224	0.799	0.936	0.975
DESC	0.149	**0.819**	**4.172**	0.221	0.805	0.934	0.975
S³Net (Test Semantic) [54]	**0.145**	0.887	4.218	**0.215**	**0.813**	**0.941**	0.977

4.4 Experiments

We discuss the experimental setup before presenting our evaluation results.

Setup. We use Pytorch 1.4 and a NVIDIA 1080 Ti GPU. We obtain the semantic annotations, in both $\mathcal{S}$ and $\mathcal{T}$, by using a ResNet-101 [122] panoptic segmentation model [163] trained on COCO-Stuff [195, 34] from the *Detectron 2* library [345]. We employ a VGG-based U-Net [274] for G_D and G_S, and a ResNet-based model for $G_{\mathcal{S}\to\mathcal{T}}$. Both image translation and depth estimation architectures are the same as the architectures used in [379, 374]. Following [374], we set $\lambda_{\mathcal{S}} = 50$, $\lambda_{GAN} = 1$, $\lambda_{Sm} = 0.01$, and following [379] we set $\lambda_{IDT} = 100$. Similarly to the original implementation of [374], we first pretrain the networks to reach good performance in $\mathcal{S}$ before introducing the consistency terms, *i.e.*, with $\lambda_{\mathcal{T}} = 0$. Afterwards, we freeze $G_{\mathcal{S}\to\mathcal{T}}$ to reduce the memory footprint, and we introduce the semantic consistency terms by setting $\lambda_{\mathcal{T}} = 1$ unless stated otherwise. The batch size is set to 4, with a 50/50 target and source data ratio, we use Adam [162] with learning rate 10^{-4} and we train for 20000 iterations after pretraining. To obtain the edge map for G_S we use a Canny Edge detector [36]. For the minimum and maximum threshold values of the Canny Edge detector we use 100 and 200 respectively, which yield good qualitative results in our datasets. We randomly change the brightness, saturation and contrast of the images for data augmentation.

Table 4.2: Ablation study of DESC for Virtual KITTI→KITTI in Eigen split [82] capped at 80m. *Img.* refers to using image translation, *Ins.* to using instance-height pseudo-labels (Section 4.3.1) and *Con.* to the consistency of predictions constraint (Section 4.3.2).

	Lower is better				Higher is better		
Method	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Only Source [374]	0.223	2.205	7.055	0.305	0.672	0.872	0.945
+ Img.	0.199	2.436	7.137	0.280	0.753	0.890	0.950
+ Img. + Con. (only edges)	0.187	1.330	6.094	0.258	0.708	0.905	0.966
+ Img. + Con.	0.173	1.235	5.776	0.244	0.748	0.919	0.969
+ Img. + Ins.	0.171	1.332	5.818	0.250	0.771	0.918	0.966
+ Img. + Ins. ($\lambda_{Sm} = 0.1$)	0.165	1.157	5.670	0.245	0.774	0.921	0.968
DESC - Full (1 h - class)	0.160	1.107	5.746	0.243	0.780	0.920	0.968
DESC - Full (unknown $f_{\mathcal{T}}$)	0.156	1.084	5.654	0.237	0.783	0.926	0.971
DESC - Full	0.156	1.067	5.628	0.237	0.787	0.924	0.970
DESC - Full (VKITTI2)	0.155	1.097	5.597	0.238	0.786	0.926	0.970
DESC - Full (R50)	0.160	1.207	6.034	0.248	0.777	0.918	0.965
DESC - Full (R50-ImageNet pre.)	**0.149**	**1.026**	**5.476**	**0.228**	**0.797**	**0.935**	**0.975**
EfficientPS Panoptic Maps	0.156	1.067	5.559	0.238	0.781	0.926	0.970
G_S - Synth	0.186	2.164	7.011	0.282	0.763	0.894	0.949
G_S - DESC	0.155	1.146	5.601	0.232	0.789	0.930	0.974
G_S - Synth + Stereo	0.136	1.206	5.598	0.235	0.822	0.932	0.969

Virtual KITTI→KITTI. For the Virtual KITTI→KITTI setting, we follow the same experimental settings as in [379, 374]. Both Virtual KITTI [92] and KITTI [97] images are downscaled to 640×192, and following [379] we cap the Virtual KITTI [92] ground truth depth at 80m (see Section 2.6.2 for further explanation of the individual datasets used).

Cityscapes→KITTI. Cityscapes [64] provides disparity maps computed using Semi-Global Matching [125]. We use the official training set, consisting of 2975 images of size 2048×1024. We set the horizon line approximately in the center by cropping the upper part, resulting in images of 2048×964. We then take the 2048×614 center crop to have the same aspect ratio as in KITTI and rescale the images to 640×192. We use $\lambda_{\mathcal{T}} = 5$ for this experiment.

Evaluation on KITTI. We follow the same evaluation protocol, metrics and splits as in [82] for KITTI, with the evaluation code from Monodepth2 [104]. The metrics used, defined in [82], are:

$$\begin{aligned} \text{Abs Rel} &= \frac{1}{|T|}\sum_{y\in T}\frac{|\hat{y}-y|}{y} \qquad \text{Sq Rel} = \frac{1}{|T|}\sum_{y\in T}\frac{(\hat{y}-y)^2}{y} \qquad \delta = \max\left(\frac{y}{\hat{y}},\frac{\hat{y}}{y}\right) < th \\ \text{RMSE} &= \sqrt{\frac{1}{|T|}\sum_{y\in T}\frac{(\hat{y}-y)^2}{y}} \qquad \log \text{RMSE} = \frac{1}{|T|}\sum_{y\in T}\frac{(\log\hat{y}-\log y)^2}{y} \end{aligned} \tag{4.5}$$

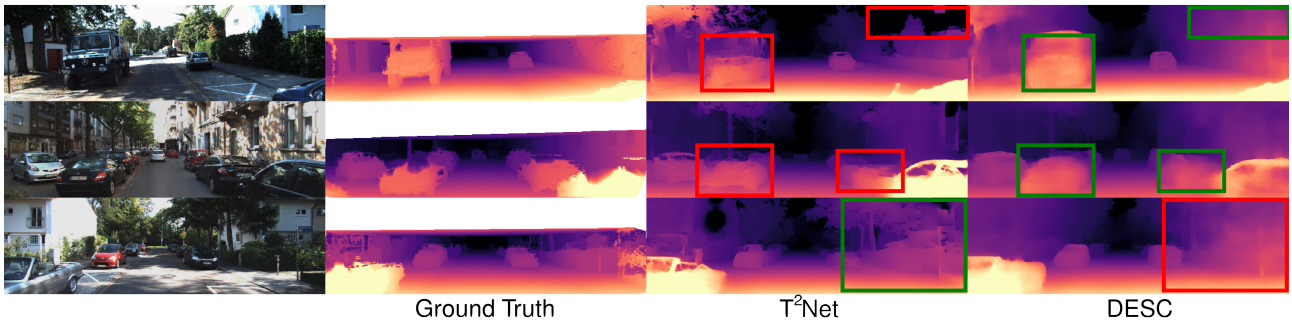


Figure 4.3: Qualitative results in KITTI for models trained on Virtual KITTI→KITTI. Ground truth depth is linearly interpolated for visualization. Green bounding boxes refer to areas of the prediction more accurate compared to the corresponding red bounding boxes.

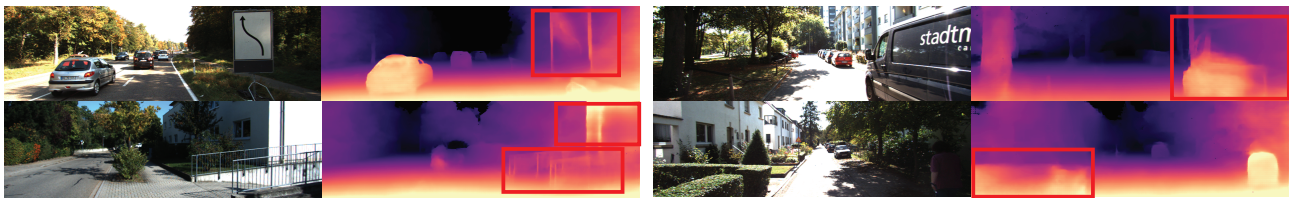


Figure 4.4: Regions with large error for our model trained on Virtual KITTI→KITTI.

where $\hat{y}$ and y are the predicted and ground-truth depth, and $|T|$ is the number of pixels with valid ground-truth for a given image. The thresholds *th* used for δ are $(1.25, 1.25^2, 1.25^3)$. These metrics are averaged for the test set, and the predictions are upscaled to the ground truth size. We provide results for both ground truth depth capped at 80m and between 1-50m as in [374, 379]. Except when using stereo supervision or in the semi-supervised regime, the results are reported using median scaling as in [233, 40, 381]. Median scaling aligns the predictions to the scale of the ground-truth by multiplying the prediction by the ratio of the ground-truth median depth and the median depth of the prediction.

4.4.1 Main Results on Virtual Kitti→Kitti

Comparison with State-of-the-Art. Table 4.1 compares the performance of DESC with the state-of-the-art Virtual KITTI→KITTI methods that do not use stereo nor video self-supervision in KITTI. DESC performs better than AdaDepth [233] and T^{2}Net [379], with a Sq. Rel. error almost 24% lower than T^{2}Net. Compared to *S^3Net* [54], which also uses semantic information during training, DESC performs better in most metrics than *S^3Net* but is outperformed by *S^3Net (Test Semantic)*, as the latter employs semantic maps during test time. The recent domain generalization method S2R-DepthNet [49] achieves comparable results to DESC without using any KITTI data during training,

Table 4.3: Results on KITTI Eigen split (80m cap) for methods using stereo data in KITTI. Due to an evaluation error in [374], results from GASDA are recomputed using the official pretrained models. We include a state-of-the-art stereo-trained method, *Monodepth2* [104].

	Lower is better				Higher is better		
Method	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Virtual KITTI→ KITTI							
Source + Stereo	0.131	1.154	5.518	0.227	0.837	0.937	0.971
T^{2}Net [379] + Stereo	0.126	1.114	5.429	0.223	0.839	0.938	0.971
GASDA [374]	0.124	1.018	5.202	0.217	0.846	0.944	0.973
DESC + Stereo	0.119	**0.935**	**5.050**	0.217	0.843	0.942	0.974
SharinGAN [256]	**0.116**	0.939	5.068	**0.203**	**0.850**	**0.948**	**0.978**
Only KITTI							
Monodepth2 (w/o pre.) [104]	0.130	1.144	5.485	0.232	0.831	0.932	0.968
Monodepth2 (ImageNet pre.) [104]	0.109	0.873	4.960	0.209	0.864	0.948	0.975

especially in the 80m cap setting. In Section 4.4.2, we explore the combination of S2R-DepthNet and DESC, which improves notably the results of DESC. Figure 4.3 shows some predictions of our DESC method compared to T^{2}Net, which we build upon. DESC contains fewer high-error regions than T^{2}Net [379] due to the guidance provided by G_S, as shown in the upper-right wall in the first row. The geometry of the instances in our method tends to be complete, *e.g.*, the cars of the second row and the larger car in the first row, which has large missing parts in the T^{2}Net prediction. The last row in Figure 4.3 also shows that DESC produces less detailed regions due to the consistency term with G_S, which blurs the predictions and removes some fine structures. Figure 4.4 includes more examples of DESC predictions with erroneous regions. Due to the overexposure present in KITTI, in some white regions (*e.g.*, traffic sign in the top-left example and wall in the bottom-left example) the model incorrectly predicts larger depth values. Even though DESC improves the completeness of the geometry of the instances, there are some instances without a fully-complete predicted depth (top-right example). Related to the loss of detail due to the consistency with G_S, DESC also blurs the predictions for thin objects (hand-rail in bottom-left example), or ignores some structures (bottom-right image).

Ablation Study. Table 4.2 shows an ablation study of DESC. The result marked with $+Img$ correspond to T^{2}Net [379] without the adversarial feature module, and with a lower smoothing weight λ_{Sm} as we use $\lambda_{Sm} = 0.01$ instead of the original $\lambda_{Sm} = 0.1$ used in *T^{2}Net* [379] in Table 4.1. The lower λ_{Sm} we use accounts for the better results of T^{2}Net in Table 4.1. We chose a smaller λ_{Sm} for

our experiments because a larger λ_{Sm} blurs the predictions, leading to a worse result after enforcing consistency with G_S due to the loss of detail. However, when consistency with G_S is not applied, a larger λ_{Sm} is beneficial as shown by the improved results of *+Img.+Ins.* ($\lambda_{Sm} = 0.1$) compared to *+Img.+Ins.*. Both the instance-based pseudo-labelling and consistency with G_S modules bring an improvement as shown in *+Img.+Ins.* and *+Img.+Con.* compared to *+Img*. Using the consistency term when only edge maps are input into G_S improves most metrics as shown in + *Img.+Con. (only edges)*, although it also shows that inputting the semantic map into G_S is largely beneficial. *DESC - Full* shows an improvement in all metrics, also compared to learning a single h per class in *DESC - Full (1 h per class)* as in Struct2Depth [40]. For *DESC - Full (unknown $f_{\mathcal{T}}$)* we set $f_{\mathcal{T}}$ to half the actual value, obtaining comparable results to when using the correct value of $f_{\mathcal{T}}$, *i.e.*, in *DESC - Full*. This result shows that ϕ in Equation (4.2) automatically scales the instance size pseudo-labels to the correct range for unknown $f_{\mathcal{T}}$.

Virtual KITTI 2 [32] is an updated version of Virtual KITTI that replicates Virtual KITTI while improving the visual quality of the synthetic images. We include in Table 4.2 the performance of DESC when using Virtual KITTI 2 as our source data. Despite the higher-quality synthetic images, the results of DESC are comparable when using either Virtual KITTI or Virtual KITTI 2, which we attribute to the image translation module outputting similar source to target translations to those when using Virtual KITTI.

ImageNet Pretrained G_D. Our G_D network is a randomly initialized VGG-based model, which was also employed in past Domain Adaptation works [379, 374]. However, some state-of-the-art depth estimation works [104] use a ResNet-based network initialized with ImageNet weights, as transfer learning has proven to be beneficial for depth estimation [5]. On that note, we run our DESC approach using for G_D the ResNet50-based network [122] with an ImageNet pretrained encoder given in the Monodepth2 [104] implementation. Compared to the VGG-based U-Net used for the base DESC, we obtain lower performance with the randomly initialized ResNet-50 as shown in *DESC - Full (R50)* in Table 4.2. However, when using an ImageNet initialized encoder, line *DESC - Full (R50-ImageNet pre.)* in Table 4.2, the absolute relative error is improved by 4.5% compared to *DESC - Full*, which highlights the importance of ImageNet pretraining also in a Domain Adaptation setting as we obtain faster training and better performance.

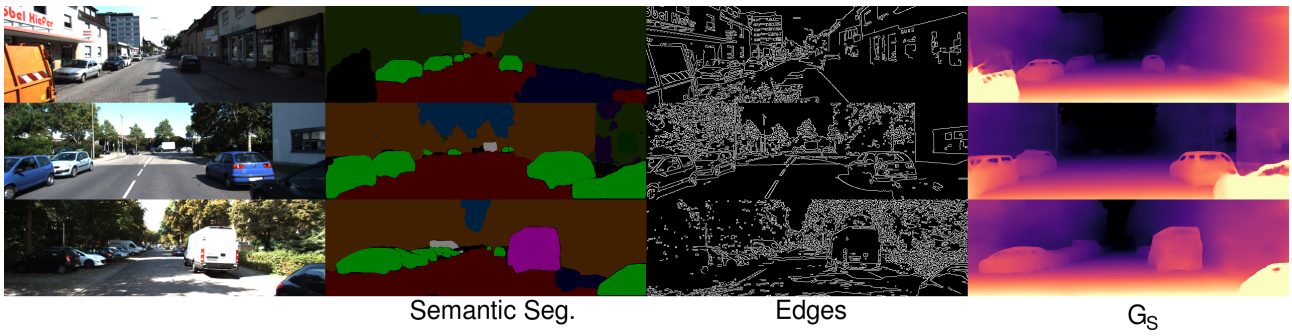


Figure 4.5: Qualitative results when using G_S, which takes as input the concatenation of a semantic segmentation (second column) and an edge map (third column) to output a depth map (last column).

Panoptic Model. For the semantic segmentation map used in G_S we leveraged a COCO-trained model using the *Detectron 2* [345] library. However, COCO includes both high diversity images and classes that are not relevant to the driving data in KITTI/Virtual KITTI. For that reason, we aim to test if using the panoptic predictions from a model trained in a more similar domain, in this case the state-of-the-art method EfficientPS [227] trained on Cityscapes [64], has an impact on the performance. Table 4.2 shows that the results are comparable for most metrics when using either Detectron2, line *DESC - Full*, or EfficientPS, in the line *EfficientPS Panoptic Maps*. This similarity of results suggests that using a panoptic segmentation model trained on a more general dataset, *i.e.*, COCO, seems to not degrade the performance compared to selecting a more domain-specific panoptic model.

Performance of G_S. Table 4.2 includes the results of G_S after being trained only with Virtual KITTI data (G_S - *Synth*), and when combining Virtual KITTI dense supervision and KITTI stereo supervision (G_S - *Synth+Stereo*). Even though G_S removes all texture information, the performance of G_S with stereo supervision is comparable to the main network G_D with stereo supervision (Table 4.3, line *Source+Stereo*). Furthermore, we argue that the improved results of G_D after applying semantic consistency (Table 4.2, line *+Img.+Con.*) compared to only using image translation (Table 4.2, line *+Img*) are not due to a distillation process, *i.e.*, due to G_S having a higher accuracy and transferring its performance to G_D after source data pretraining. Instead, the lower performance of G_S - *Synth* compared to both G_D after applying semantic consistency (*+Img.+Con.*) and G_S after full training (G_S - *DESC*), suggests that consistency between predictions from different modalities constrains the learning process and is the reason for the accuracy increase, as both G_D and G_S models benefit from the consistency loss term. In Figure 4.5 we show some examples of predictions of our G_S module.

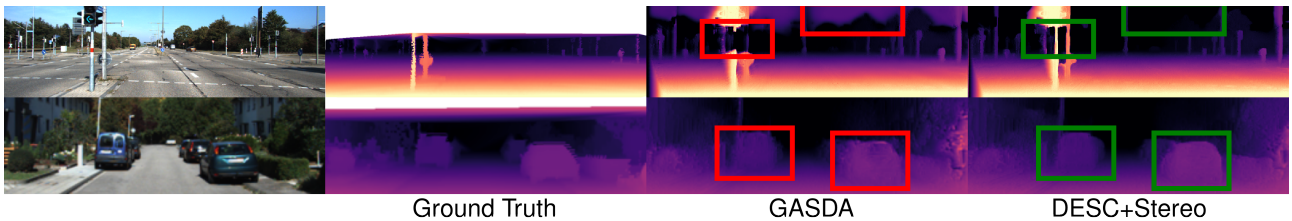


Figure 4.6: Qualitative results in KITTI for models trained on Virtual KITTI→KITTI with stereo supervision in KITTI. Bottom row corresponds to a center crop of the original image.

The module is highly guided by the semantic segmentation mask, where errors in the prediction (*e.g.*, missing part on the left van in the top image, and non-straight edges on the bottom right car in the middle row) translate to errors in the prediction. The edge map does help recover some details, *e.g.*, the car windows, but overall the resulting predictions are smooth (*e.g.*, foliage in the bottom row prediction) due to the difficulty of predicting high-frequency changes from edges and semantic maps.

Stereo Supervision. Although DESC focuses on the setting where no self-supervision is used, our approach can also bring an improvement in such a scenario. We train DESC with KITTI stereo supervision by adding the same multi-scale pixel-wise reconstruction method as GASDA [374] with the same weight of $\lambda_{St} = 50$. To account for the introduced supervision in $\mathcal{T}$, we increase $\lambda_{\mathcal{T}} = 5$ and the number of training iterations to 100,000. Table 4.3 shows that, compared to *T*^{2}*Net+Stereo*, our method with stereo supervision, *DESC + Stereo*, achieves better results in all metrics and also outperforms GASDA [374] in most metrics. GASDA is a Domain Adaptation method tailored for stereo supervision that uses two depth estimation networks and an image-translation network during inference. The recent stereo-focused Domain Adaptation method SharinGAN [256] performs better in most metrics than *DESC + Stereo* due to the improved image transfer strategy used, although SharinGAN also increases the computational cost at test time due to using an extra network. We also report better performance than the state-of-the-art for stereo supervision, Monodepth2 [104] without ImageNet [71] pretraining in *Monodepth2 (w/o pre.)*. However, ImageNet pretraining has a large effect on the accuracy of Monodepth2, shown in *Monodepth2 (ImageNet pre.)*, achieving better results than our method. Figure 4.6 shows predictions for Domain Adaptation methods using stereo supervision in KITTI. Compared to GASDA, we observe a better recovery of fine structures, shown in the pole of the first row of Figure 4.6, and better predictions of further object instances, shown in the bottom row. As shown in the first row of Figure 4.6, DESC also predicts a better depth for the sky due to the explicit supervision with synthetic depth.

Table 4.4: Results on KITTI Eigen split (cap 80m) when varying either the weight of the semantic consistency loss $\lambda_{\mathcal{T}}$, and the number of training iterations for the last step of training. Default values used in DESC are $\lambda_{\mathcal{T}} = 1.0$ and 20000 training iterations.

	Lower is better			
Method	. Abs Rel	Sq Rel	RMSE	RMSE log
Variation of $\lambda_{\mathcal{T}}$				
$\lambda_{\mathcal{T}} = 1.0$	**0.156**	**1.067**	**5.628**	**0.237**
$\lambda_{\mathcal{T}} = 2.0$	0.158	1.081	5.816	0.243
$\lambda_{\mathcal{T}} = 5.0$	0.162	1.143	6.044	0.249
Train. It. ($\lambda_{\mathcal{T}} = 1$)				
10000	0.158	1.105	**5.557**	**0.236**
20000	**0.156**	**1.067**	5.628	0.237
40000	0.157	1.077	5.661	0.239
80000	0.159	1.099	5.784	0.244

Hyperparameter Selection. The weights associated with the image translation process were chosen following T^{2}Net values, however we still need to tune the semantic consistency weight $\lambda_{\mathcal{T}}$. On that note, we include in Table 4.4 the effects of varying $\lambda_{\mathcal{T}}$. Multiplying the original $\lambda_{\mathcal{T}} = 1$ by 5, *i.e.*, line $\lambda_{\mathcal{T}} = 5$, the absolute relative error degrades by 4%, although it still achieves better performance than T^{2}Net (Table 4.1). Table 4.4 also shows the results when varying the number of training iterations when applying the last semantic consistency step of training. Even though modifying the training iterations impact the performance, the error variation is less pronounced compared to modifying $\lambda_{\mathcal{T}}$. In practice, most Unsupervised Domain Adaptation methods use quantitative performance in target data to tune to some level the hyperparameters or the model. Further research needs to develop reliable methods to avoid the use of target domain ground truth for hyperparameter or model selection [363].

Distribution of Errors. Following [116], we now analyze the error for different depth ranges and semantic classes for *DESC* and *DESC+Stereo*. As DESC builds upon T^{2}Net, we also include it to better understand the performance improvement given by DESC. Figure 4.7 shows the distribution of errors depending on the ground truth value. KITTI concentrates most of the ground truth values around 5-20 meters, thus the better performance of all methods in the 5-20 meters region is likely due to the larger amount of training data in that range. DESC consistently performs better than T^{2}Net for depth values under $\approx$ 65m, however the performance of DESC drops for depth values close to 80m. Adding stereo information to DESC improves the performance largely for closer depth values (where there are large disparity shifts) and for larger depth values, whereas it achieves similar performance

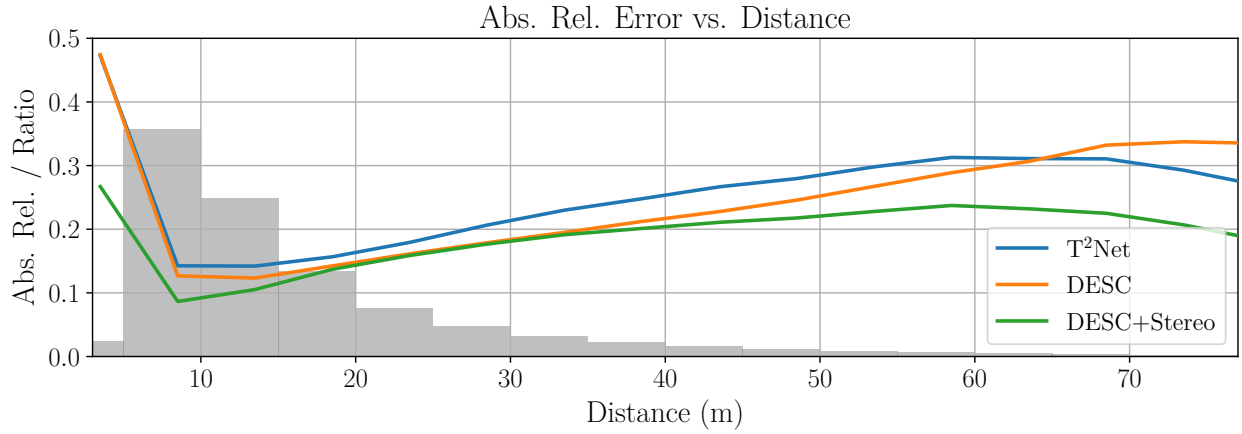


Figure 4.7: Absolute relative error vs. ground truth distance for three different methods in the KITTI Eigen test split. We also include a histogram of the ground truth distribution. Left axis shows both absolute relative error (line plot) and ratio of ground truth values (histogram).

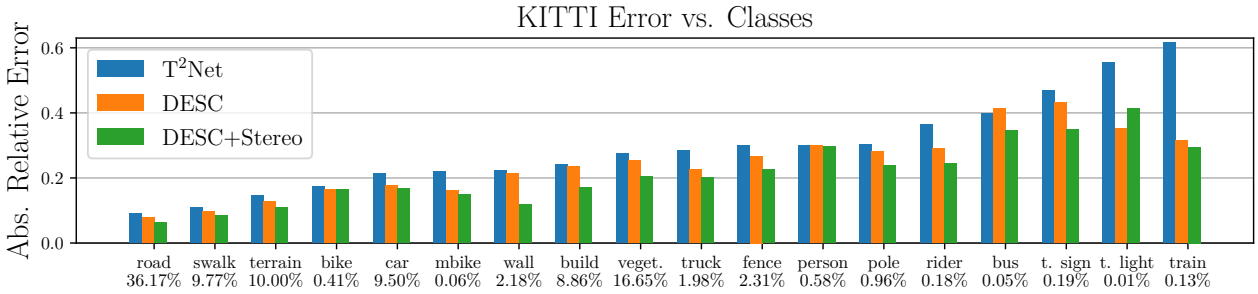


Figure 4.8: Distribution of absolute relative error averaged over all pixels for different classes over the KITTI Eigen test split (cap 80m) for three different methods trained on Virtual KITTI→KITTI. The percentage of ground truth depth corresponding to each class is given below each class.

to regular DESC for mid-range values. Figure 4.8 shows the performance for the different semantic classes (we use the EfficientPS semantic maps trained on Cityscapes for evaluation). DESC performs better than T^{2}Net for most classes, especially for the detected object instances (*e.g.*, car, train or traffic light) due to the semantic consistency introduced in DESC, which is also related to the better completeness of instances in Figure 4.3.

4.4.2 Improved Image Transfer Strategies

The image transfer approach used in DESC [209] is based upon T^{2}Net [379]. However, recent Domain Adaptation and generalization works outperformed the image transfer method in T^{2}Net. We now aim to substitute the T^{2}Net strategy used in DESC with these improved image transfer strate-

Table 4.5: Results of DESC on the KITTI Eigen test split with different image transfer modules.

Method	Lower is better				Higher is better		
	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Cap 80m							
DESC - T^{2}Net [379]	0.156	1.067	5.628	0.237	0.787	0.924	0.970
DESC - SharinGAN [256]	0.153	1.057	5.641	0.236	0.789	0.926	0.970
DESC - S2R-DepthNet [49]	**0.145**	**0.986**	**5.368**	**0.225**	**0.806**	**0.938**	**0.976**
Cap 50m							
DESC - T^{2}Net [379]	0.149	0.819	4.172	0.221	0.805	0.934	0.975
DESC - SharinGAN [256]	0.146	0.804	4.123	0.219	0.807	0.937	0.975
DESC - S2R-DepthNet [49]	**0.139**	**0.742**	**3.971**	**0.211**	**0.821**	**0.947**	**0.980**

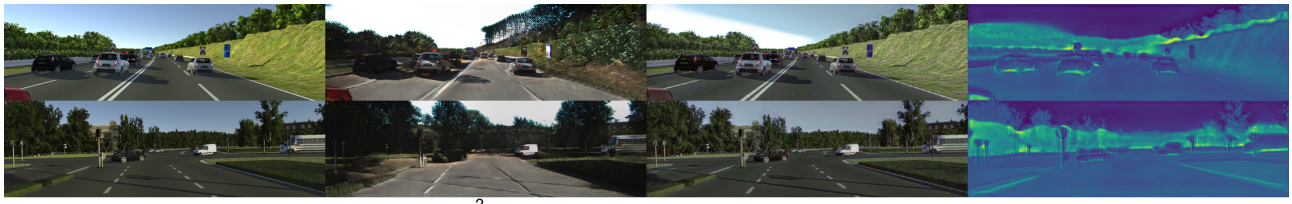


Figure 4.9: Qualitative results for three different image transfer strategies. Virtual KITTI images are transferred to either a style matching KITTI images using a T^{2}Net-based approach or to an intermediate shared domain with the transferred KITTI images using either a SharinGAN [256] or S2R-DepthNet [49] approach. The original single-channel S2R-DepthNet images are mapped to RGB using a colormap and logarithmic mapping.

gies. Specifically, we include in DESC the methods presented in SharinGAN [256] and the domain generalization method S2R-DepthNet [49] (see Section 2.3). To apply SharinGAN to DESC we use the image transfer network pretrained on Virtual KITTI→KITTI from the official SharinGAN code and keep it frozen during training. For S2R-DepthNet, we take the official pretrained models, which are trained on Virtual KITTI, and further finetune both the depth network and the attention network using our semantic consistency modules. For both SharinGAN and S2R-DepthNet approaches, we translate both target and source data to an intermediate domain before feeding the images to G_D.

Quantitative Results. Table 4.5 shows the results for the three image transfer approaches used with DESC. Both SharinGAN and S2R-DepthNet further improved the performance of DESC. Using SharinGAN instead of a T^{2}Net approach lowers the absolute relative error in cap 80m by 2%, whereas using S2R-DepthNet has a larger impact on the DESC performance, decreasing the absolute relative error by 7%. *DESC - DepthNet* outperforms in all metrics both the base S2R-DepthNet and base DESC results given in Table 4.1. The increased performance achieved when combining DESC with either method also shows the wide applicability of DESC to other image translation methods.



Figure 4.10: Transfer of KITTI images to the intermediate shared domain employed by SharinGAN [256] and S2R-DepthNet [49]. The original single-channel S2R-DepthNet images are mapped to RGB using a colormap and logarithmic mapping.

Qualitative Results. Figure 4.9 shows examples of image translations for our trained T^{2}Net approach, the SharinGAN model and the depth structure output by S2R-DepthNet. T^{2}Net produces a stronger shift in the Virtual KITTI images compared to SharinGAN, resembling more the illumination and textures present in the real KITTI at the cost of introducing artifacts (*e.g.*, hallucinated trees). Furthermore, Figure 4.9 and Figure 4.10 show that the SharinGAN translations for both Virtual KITTI and KITTI images are quite close to the input image, suggesting that non-aggressive changes are enough to achieve good performance. S2R-DepthNet produces images quite different to those from either T^{2}Net or SharinGAN. S2R-DepthNet mostly removes texture and illumination cues (*e.g.*, no shadows in the cars of Figure 4.9) and only keeps the structural edges needed for depth prediction. Hence, S2R-DepthNet maps the input RGB images to a lower-gap intermediate domain, as shown in Figure 4.9 and Figure 4.10 where the resulting S2R-DepthNet images from KITTI and Virtual KITTI are quite closer in appearance compared to the original RGB images.

Computational Complexity. We use as GPU a NVIDIA 1080 Ti. DESC only adds computational cost during training, hence the inference speed depends on the network used. The U-Net employed in DESC for G_D, also used in T^{2}Net and GASDA, is capable of an inference of 43 imgs/s with a resolution of 640×192, assuming a batch size of 1. Using the presented improved image-transfer strategies reduces the inference speed, as both SharinGAN and S2R-DepthNet approaches need extra networks at test time. In the case of using a *DESC-SharinGAN* approach, the inference speed decreases to 23 imgs/s, whereas with *DESC-S2R-DepthNet* we achieve a speed of 9 imgs/s. The total training time for *DESC - T^{2}Net* is approximately 2 days, which accounts for the three training steps (*i.e.*, pretraining of G_D, pretraining of G_S and joint training) and assumes the panoptic predictions are precomputed.

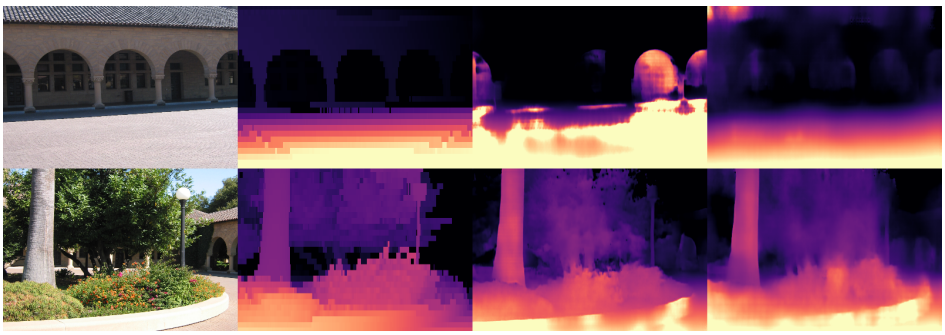


Figure 4.11: Qualitative results in Make3D for T^{2}Net and DESC.

Table 4.6: Results on Make3D (70m cap) [298] using the same central image crop as [256]. All of the methods have been trained on Virtual KITTI→KITTI, except for S2R-DepthNet, which is only trained on Virtual KITTI. For non-median scaled results of SharinGAN, GASDA and T^{2}Net, we report the results given in [256], and we compute the median scaled results using the evaluation code given in [256] along with the official pretrained models given by each method.

	Lower is better					
	No Median Scaling			Median Scaling		
Method	Abs Rel	Sq Rel	RMSE	Abs Rel	Sq Rel	RMSE
S2R-DepthNet [49]	0.490	10.676	10.889	0.485	10.547	10.833
T^{2}Net [379]	0.508	6.589	8.935	0.420	7.477	9.992
GASDA [374]	0.403	6.709	10.424	0.377	6.323	9.097
SharinGAN [256]	**0.377**	4.900	8.388	0.322	3.744	7.812
DESC	0.424	5.563	8.571	0.335	3.772	8.030
DESC (R50-ImageNet pre.)	0.386	**3.943**	**8.104**	**0.293**	**2.755**	**7.510**

4.4.3 Evaluation in Additional Settings

Make3D [298] is used to test the generalization capabilities of our DESC model trained in Virtual KITTI→ KITTI. The ground truth in Make3D is of low quality and low resolution, as shown in the examples in Figure 4.11, hence the results provide only rough guidance of the generalization ability of the model. We use the evaluation protocol and code given by another method, SharinGAN [256], for a fair comparison. We include in Table 4.6 the results of both our original DESC and our DESC with ImageNet pretraining. We also report the results on Make3D both with and without median scaling, as median scaling greatly improves the results due to the different camera intrinsics and image resolution in Make3D affecting the scale of the predictions. Table 4.6 shows that our method performs comparatively well in Make3D, and the only Domain Adaptation method that obtains similar results is SharinGAN [256], which contrary to DESC uses stereo information from the real domain during training. Compared to T^{2}Net [379] and S2R-DepthNet, which are the other methods in Table 4.6 that

Table 4.7: Results on the KITTI Eigen test split (cap 80m) when using a semi-supervised setting with 1000 labeled KITTI images. No median scaling performed during evaluation for this experiment. Results for *T²Net*, *Real + Syn* and *ARC* taken from [378].

	Lower is better				Higher is better		
Method	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
AdaDepthS [233]	0.167	1.257	5.578	0.237	0.771	0.922	0.971
Real + Syn	0.152	0.988	4.751	0.257	0.784	0.918	0.966
T²Net [379]	0.151	0.993	4.693	0.253	0.791	0.914	0.966
ARC [378]	0.143	0.927	4.679	0.246	0.798	0.922	0.968
DESC - Only Img. Trans.	0.132	0.995	5.085	0.215	0.824	0.937	0.976
DESC	0.128	0.924	4.984	0.210	0.829	0.940	0.977
S2R-DepthNet [49]	**0.116**	**0.766**	**4.409**	**0.185**	**0.858**	**0.955**	**0.984**

do not use any real-domain stereo supervision during training, *DESC* achieves better performance by a wide margin. Figure 4.11 shows some qualitative results on Make3D, where we observe that the predictions of both DESC and T²Net contain large areas of error, highlighting the need for methods capable of better generalization. The top row corresponds to an example where both methods fail to predict a satisfactory depth for the building, showing these generalization issues. The bottom row shows how our method, although quantitatively behaves noticeably better than T²Net, produces blurrier predictions in Make3D as a consequence of the consistency loss with G_S used during training.

Semi-supervised Setting. Past work [378, 233, 49] assumed access to 1000 KITTI images in a semi-supervised setting, which we now investigate. We use the same 1000 labeled KITTI frames as in ARC [378] as our annotated data. We finetune our final DESC model with the labeled real images following the same loss given in Equation (4.4) with the addition of the KITTI ground truth supervision loss. For the target data supervision, as the ground truth is sparse, we upscale the feature maps instead of downscaling the ground truth to leverage all of the available depth values. Table 4.7 shows that we obtain better results for *DESC - Only image translation*, which is T²Net without feature adaptation, compared to those reported in [378], which could be partially due to using a different implementation for the target loss. Table 4.7 also shows that DESC outperforms all of the past Domain Adaptation methods, but obtains lower performance than the domain generalization method S2R-DepthNet, which can be quickly adapted to new domains with few examples.

Evaluation on KITTI Stereo. KITTI Stereo 2015 [223] provides images annotated in a process combining (1) static background retrieval via egomotion compensation and (2) fitting of CAD models to

Table 4.8: Results on the KITTI 2015 stereo 200 training set disparity images [223, 97]. We include *Monodepth2* [104], the state-of-the-art stereo method trained only in KITTI. Results for non-stereo trained methods (*T²Net*, *S2R-DepthNet* and *DESC*) are reported with median scaling.

	Lower is better				Higher is better		
Method	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Virtual KITTI→KITTI							
T²Net [379]	0.151	1.535	6.177	0.224	0.817	0.935	0.975
S2R-DepthNet [49]	0.142	1.371	5.737	0.207	0.835	0.949	0.981
DESC	0.120	0.968	5.597	0.206	0.839	0.937	0.977
GASDA [374]	0.095	1.068	5.015	0.168	0.906	0.966	0.986
SharinGAN [256]	0.092	0.903	4.611	0.159	0.906	**0.968**	**0.987**
DESC + Stereo	**0.085**	**0.781**	**4.490**	**0.158**	**0.909**	0.967	0.986
Only KITTI							
Monodepth2 (w/o pre.) [104]	0.096	1.163	5.161	0.179	0.898	0.959	0.981
Monodepth2 (ImageNet pre.) [104]	0.082	0.908	4.698	0.158	0.919	0.970	0.986

Table 4.9: Cityscapes→KITTI results, evaluated in KITTI [97] Eigen split (80m cap). *Struct2Depth (M+R)* [40] uses three consecutive frames for refinement.

	Lower is better				Higher is better		
Method	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Only Cityscapes							
Source Baseline	0.189	1.717	6.478	0.257	0.740	0.919	0.968
Struct2Depth (M) [40]	0.188	1.354	6.317	0.264	0.714	0.905	0.967
Struct2Depth (M+R) [40]	0.153	1.109	5.557	0.227	0.796	0.934	0.975
Cityscapes→ KITTI							
T²Net [379]	0.173	1.335	5.640	0.242	0.773	0.930	0.970
DESC (Img.+Ins.)	0.174	1.480	5.920	0.240	0.782	0.931	0.971
DESC (Img.+Con.)	0.150	0.981	5.359	**0.222**	0.805	0.938	**0.976**
DESC (Full, $\phi = 1$)	0.169	1.142	5.936	0.261	0.741	0.919	0.967
DESC (Full, ϕ learned)	**0.149**	**0.967**	**5.236**	0.223	**0.810**	**0.940**	**0.976**

account for dynamic objects. The result is a denser ground truth compared to the LiDAR annotations provided in KITTI, especially in the cars. DESC, which uses detected instances to generate depth pseudo-labels, benefits from evaluating in images with denser annotation in the vehicles, as shown in Table 4.8 in the larger accuracy gap between *DESC* and *T²Net*, and between *DESC* and *S2R-DepthNet* compared to the results on the Eigen split given in Table 4.1. Comparing stereo-trained methods we find a similar trend, there is a larger gap in performance between *DESC + Stereo* and *GASDA*, and *DESC + Stereo* also outperforms *SharinGAN* contrary to the results in Table 4.3. Furthermore, *DESC + Stereo* achieves either better (Sq Rel, RMSE) or equal (RMSE log) squared metrics results than the state-of-the-art *Monodepth2 (ImageNet pre.)* without pretraining G_D in ImageNet.

Cityscapes→ KITTI. Table 4.9 shows the results for this benchmark. We improve upon T²Net for

all metrics, with a 13.9% lower absolute relative error. Most of the accuracy improvement comes from the consistency term as shown in *DESC (Img.+Con.)* and *DESC (Full, ϕ learned)*. Due to the camera difference between the datasets, the learnable scalar ϕ is necessary for good performance, as shown for fixed $\phi = 1$ in *DESC (Full, ϕ=1)*. Struct2Depth [40] also uses precomputed semantic annotations to improve its self-supervised video learning, although Struct2Depth is not a Domain Adaptation method as it only trains with Cityscapes [64] data, *i.e.*, it does not use KITTI for training. Struct2Depth also uses a different crop for Cityscapes. Table 4.9 shows that we achieve better accuracy than *Struct2Depth (M+R)*, which uses three frames at test time for refinement, whereas we only need a single image for inference.

4.4.4 Limitations

Due to the consistency term with G_S, our method shows some loss of detail in fine structures compared to T^{2}Net [379], as shown in the last row of Figure 4.3 or in Figure 4.11, which could also limit the achievable upper-bound performance in settings with real data supervision, such as self-supervision or semi-supervised settings. Additionally, DESC is more computationally demanding during training than T^{2}Net due to the added G_S. The depth predicted by G_S also relies on the quality of the computed semantic data, hence in settings where the extracted annotations are of low quality the performance of the method may degrade. Furthermore, the instance-based pseudo-labelling predicts a height that assumes that the object is in an upright position, thus some rotations of the camera poses or objects could degrade the performance of that module.

4.5 Conclusion

We proposed a method that leverages semantic annotations to improve the performance of a depth estimation model in a Domain Adaptation setting. We used the relationship between instance size and depth to provide pseudo-labels in the target domain. A segmentation map and an edge map were input to a second network, whose prediction was forced to be consistent with the prediction of the main network. These additions led to higher accuracy in the setting where no self-supervision is available

in the real data. In the Virtual KITTI to KITTI benchmark we outperform all of the other methods that do not use KITTI video or stereo supervision, and when employing a more advanced image strategy, we also outperform a method using semantic labels at test time. As we use automatically extracted semantic annotations, our method can be easily added to current approaches to improve their accuracy in a Domain Adaptation setting, as shown in the improvement achieved with stereo self-supervision or the multiple image-transfer strategies we successfully test. As future work, approaches aiming to reduce the detail loss due to the enforced consistency of predictions could improve the method.

Chapter 5

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

The acquisition of dense ground truth annotations for depth completion is challenging, and synthetic data can provide us with pixel-perfect dense depth map (see Section 2.4 for depth completion background). However, naive training with synthetic data achieves a limited performance in a LiDAR+RGB depth completion setting. In this chapter we propose an explanation for the performance gap in a depth completion benchmark when training with synthetic data, and we validate it by generating a synthetic dataset based on the proposed assumption that improves the domain gap. In addition, we present a method to obtain reliable pseudo-labels from the real domain. The combination of these modules improves upon any past method that does not use real data ground-truth.

5.1 Introduction

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 to obtain a dense depth map 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 [213, 333, 258, 211, 357, 353]. Depth ground truth annotations are hard to acquire, being more difficult to obtain in a depth completion task due to the need of having a denser (and less noisy) depth map than the input depth. The most prominent publicly available data for this task [331] 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 [331]. To avoid such annotations, some methods perform self-supervision [211, 357, 343], 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 [275, 92, 78] with perfect ground truth, but that a domain gap usually prevents from successful training on synthetic data only.

Contributions and Outline. Contrary to past self-supervised methods [211, 357, 343], we propose to use a Domain Adaptation approach to address the depth completion problem without real data ground truth as shown in Figure 5.1. We train our method from the synthetic data generated with the driving simulator CARLA [78] (see Section 2.6.2 for more details about CARLA), and evaluate it on the real KITTI depth completion benchmark [331]. The real LiDAR data is noisy, with the main source of noise being the see-through artifacts that occur after projecting the LiDAR point cloud to the RGB cameras. To reduce the domain gap, 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 (shown in Figure 5.1), employing random masks from the real LiDAR to sparsify the virtual LiDAR map, 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 to obtain a set of reliable points that are used as supervision. Finally, to reduce the domain gap between the RGB images, we use a CycleGAN [382] to transfer the image style from the real domain to the synthetic one.

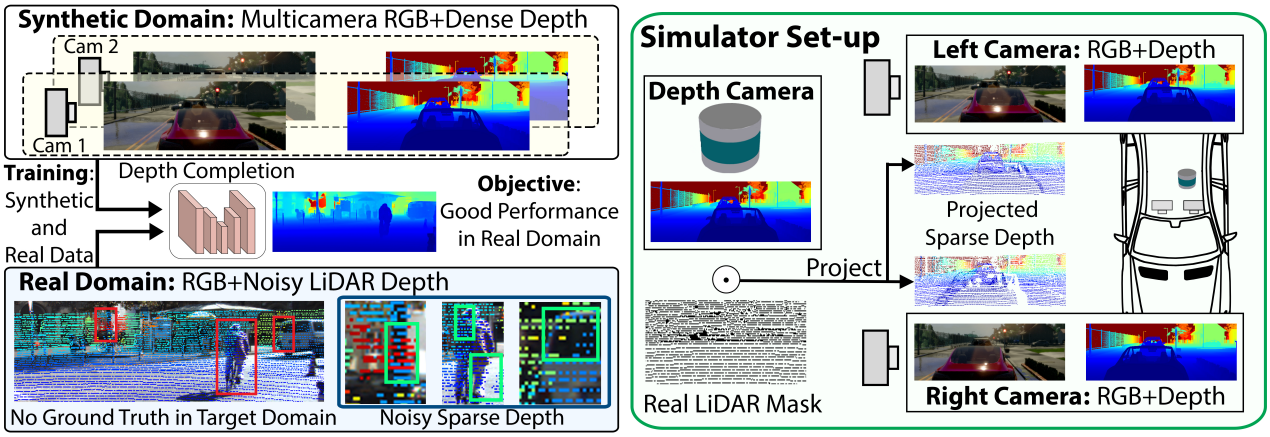


Figure 5.1: Left: We investigate the depth completion problem without ground truth in the real domain, which contains paired noisy and sparse depth measurements and RGB images. We highlight some noise present in the real data: see-through on the tree trunk and bicycle, self-occlusion on the bicycle and missing points on the van. We leverage synthetic data with multicamera dense depth and RGB images. Right: Overview of the multicamera set-up in CARLA used to simulate the real projection LiDAR artifacts. 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.

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 in the KITTI depth completion benchmark. To this end, our main contributions are:

- We use a simulator to collect and release a multi-camera synthetic dataset mimicking the set-up of a well-known real dataset, KITTI [97].
- We provide an explanation for the reduced performance on real data when naively training on synthetic data. Based on this observation, we develop an approach for geometric and data-driven sensor mimicking aimed to generate realistic input sparse depth.
- We show that RGB adaptation is secondary compared to input depth adaptation.
- Naive pseudo-labelling (*i.e.*, using input depth as supervision) limits the achievable performance. We thus propose a method for input data filtering to boost the performance.

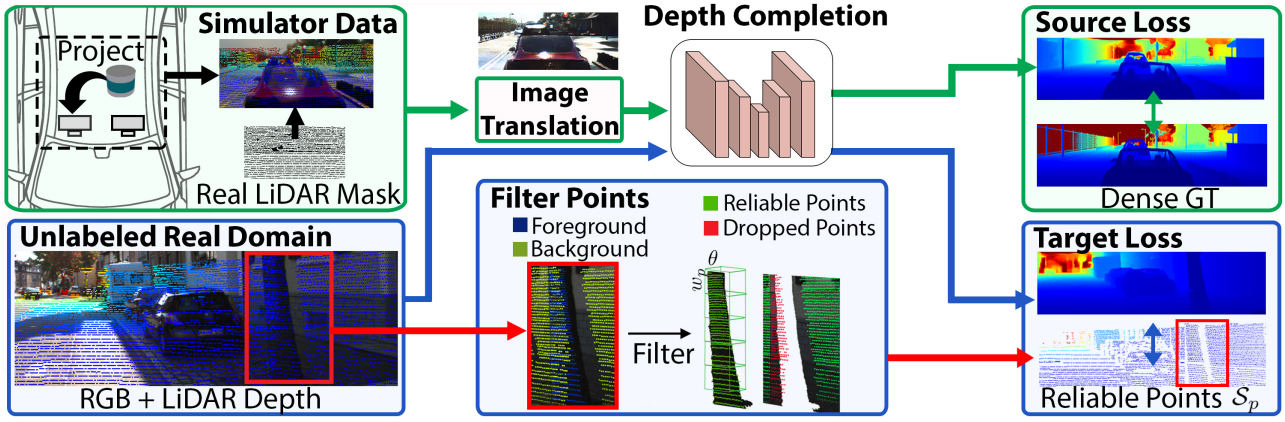


Figure 5.2: Overview of our method. We use a simulator with a multicamera set-up and real LiDAR binary masks to transform the synthetic dense depth map into a noisy and sparse depth map. The method has two training steps: green blocks are used in the 1st and 2nd step of training while blue blocks in the 2nd step only. In the filtering block, green points are the reliable points $\mathcal{R}_p$ and red points are dropped. The *Image Translation* network is pretrained using a CycleGAN approach [382].

5.2 Related Work

We now introduce past depth completion works that, similarly to our method, avoid the need of real ground-truth by leveraging self-supervision (see Section 2.4 for the relevant literature on Domain Adaptation for depth completion, and for depth completion models).

Self-Supervised Approaches. Extra views either from a second camera or a video sequence can be used for self-supervision by leveraging photometric losses. Contrary to depth estimation approaches, the input depth (along local matching methods) can be used directly to compute the pose between two frames as in [211], where a photometric loss was used on the reprojected image. Conversely, the camera poses between two frames can also be computed using a separate pose network that leverages only RGB information as in [343]. Furthermore, Wong *et al.* [343] combined video self-supervision with a two-stage approach that, first, uses a non-learning method to form a spatially dense but coarse depth approximation from the sparse points, which is then refined using another network. Stereo supervision can also be employed, and it was integrated in the probabilistic formulation proposed by Yang *et al.* [357] with a conditional prior within a maximum a-posteriori estimation. We argue that self-supervision, when available, can be integrated into methods that aim to leverage synthetic data. On that note, we will show improved results when combining our Domain Adaptation method with the temporal self-supervised approach proposed in [211].

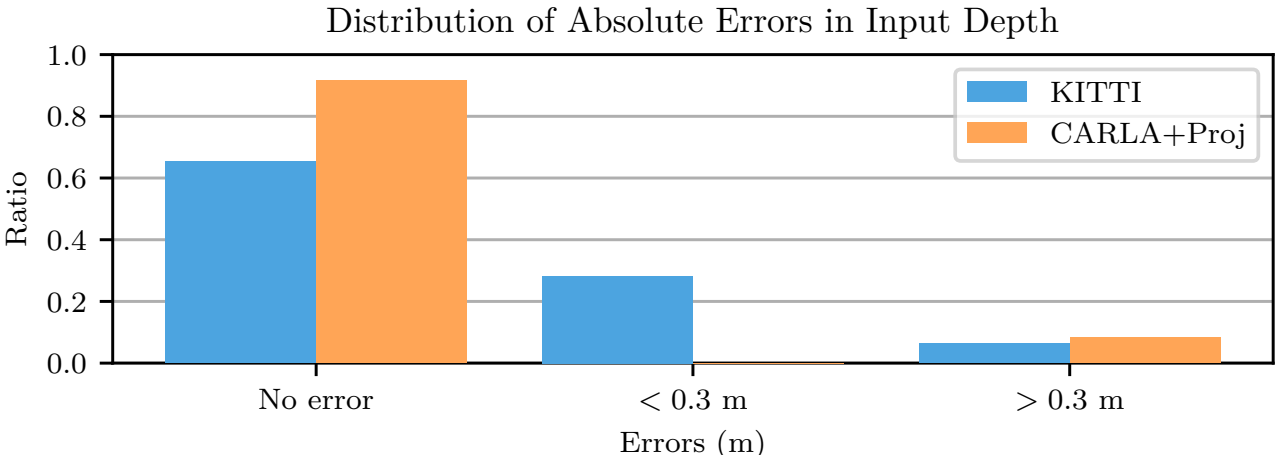


Figure 5.3: Distribution of errors in the input depth for the KITTI training set and for our collected CARLA dataset with our projection module.

5.3 Method

Our method, shown in Figure 5.2, 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.

5.3.1 Data Generation via Projections

Supervised depth completion methods strongly rely on the sparse depth input, achieving good performance without RGB information [211, 333]. 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 [364], 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 [137, 145, 357]. To generate a sparse depth $x_D^{s_B}$, a Bernoulli distribution per pixel is used in some works [137, 145, 213] 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

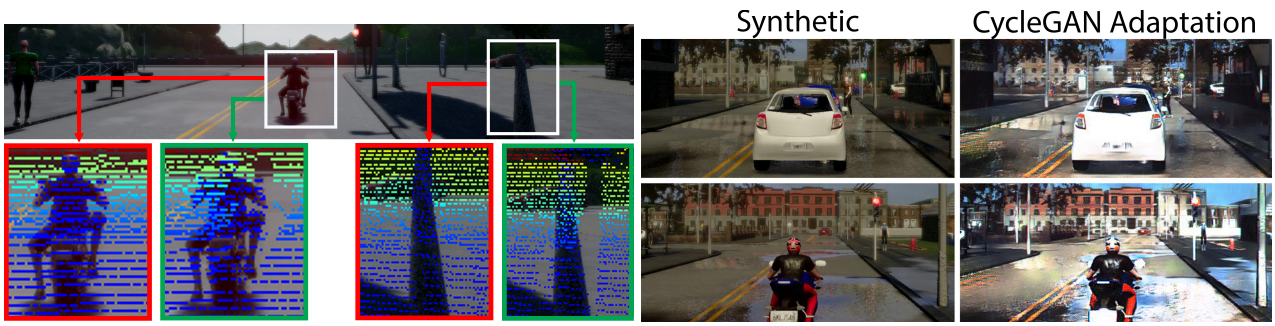


Figure 5.4: Left: Example of the generated projection artifacts in the simulator. The zoomed-in areas marked with red rectangles correspond to $x_D^{s_M}$ and the zoomed-in areas marked with green rectangles to $x_D^{s_P}$, where we can see simulated projection artifacts, *e.g.*, see-through points on 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.

to 0 with probability $(1-p_B)$. Furthermore, in a Bernoulli setting each pixel is sampled independently from each other. We argue that a model trained with $x_D^{s_B}$ does not perform well in the real domain, and our results in Section 5.4 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, in the Bernoulli sparsified depth $x_D^{s_B}$, there is no noise in the sampled points, as we directly sample from the ground truth. Regarding the noise in the input data, Figure 5.3 shows the error distribution in the input sparse data in the KITTI dataset. Most of the LiDAR sparse depth ($\approx 65\%$) contains no noise, whereas the amount of points with an error of less of 30 cm amounts to $\approx 28\%$ of the data. However, the problematic points are those 7% of input sparse points that contain an error larger than 30 cm, where the errors can be of tens of meters. These input sparse points with a large amount of error are the points that we aim to simulate in our data in order to train a model capable of correcting them.

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 [11]. 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 using a Hadamard product between the dense depth and the binary mask, $x_D^{s_M} = M_L \odot x_D$. By using sampling masks directly from the real data, we adapt the synthetic data to the sparsity level in the real domain without the need to tune any parameter depending on the LiDAR device used.

Generating Projection Artifacts. Previous works use noise-free sparse data to pre-train [258] or evaluate a model [145] 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 that setup, the error is a result of projecting a sparse point cloud arising from another viewpoint, as 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 5.4, which is also observed in the real domain [57]. 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 a simulator, CARLA [78], which allows us to capture multicamera synchronized synthetic data. Our CARLA set-up mimics the KITTI [97] camera distances, as our benchmark is the KITTI depth completion dataset [331]. Instead of a LiDAR, we use a virtual dense depth camera. The set-up is illustrated in Figure 5.1. 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

$$x_D^{s_P} = K_{RGB} P_{RGB}^{L} K_L^{-1} x_D^{s_M} \tag{5.1}$$

where K_L, K_{RGB} are the LiDAR and RGB intrinsics, P_{RGB}^{L} is the rigid transformation between the LiDAR and RGB reference frame, $x_D^{s_P}$ is the projected sparse input to either left or right camera, and for simplicity of notation we omitted the homogenization/dehomogenization process of the image coordinates. Figure 5.3 shows the error distribution in the input points after the projection step. Compared to the real data, our projection module generates a comparable amount of depth values with an error larger than 30 cm. Figure 5.3 also shows that our projection module does not simulate

the low-scale errors, *i.e.*, those with an error lower than 30 cm, which are due to other sources of error in the LiDAR data.

5.3.2 RGB Adaptation

Similarly to Domain Adaptation for depth estimation methods [379, 10, 374], 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 [382] 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 5.4.

5.3.3 Filtering Projection Artifacts for Supervision

In a depth completion setting, the given LiDAR depth can also be used as supervision data, as in [211]. The approach in [211] 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, as shown in Figure 5.3, 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 [57] 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 $\mathcal{R}_p$, likely to be correct, for supervision in the real-domain based on the assumption used in Section 5.3.1, *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 5.2. The points from the closer plane are more likely to be correct as part of the occluding objects. To retrieve $\mathcal{R}_p$ we apply a minimum pooling with window size w_p yielding a minimum depth value d_m per window. We then include in $\mathcal{R}_p$ 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 [182, 118, 372], to select w_p and θ in the synthetic validation set, thus not requiring any ground truth in the real domain. Section 5.4 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.

5.3.4 Summary of Losses

Our proposed loss is

$$\mathcal{L} = \lambda_{\mathcal{S}} \mathcal{L}_{\mathcal{S}} + \lambda_{\mathcal{T}} \mathcal{L}_{\mathcal{T}} \tag{5.2}$$

where $\mathcal{L}_{\mathcal{S}}$ is the loss used for the source synthetic data, $\mathcal{L}_{\mathcal{T}}$ the loss used for the target real data and $\lambda_{\mathcal{S}}$ and $\lambda_{\mathcal{T}}$ are hyperparameters.

Similar to past Domain Adaptation works using pseudo-labels [386, 322], we use a two-step training approach, aiming first for good performance in the synthetic data before introducing noise in the labels. First, we set $\lambda_{\mathcal{S}} = 1.0$ and $\lambda_{\mathcal{T}} = 0.0$, to train only from the source synthetic data. For $\mathcal{L}_{\mathcal{S}}$ we use the Reverse Huber (BerHu) loss [387], which has been used in past depth estimation works [170]. BerHu behaves like a Mean Squared Error (MSE) loss for predictions with larger errors and like a Mean Absolute Error (MAE) loss for predictions closer to the ground-truth. As depth-related metrics are usually a mix between l_1 and l_2-based metrics, BerHu usually leads to good results in all metrics [85]. Hence, we define $\mathcal{L}_{\mathcal{S}}$ as

$$\mathcal{L}_{\mathcal{S}} = \frac{1}{b_{\mathcal{S}}} \sum_i \frac{1}{n_i} \sum_k \mathcal{L}_{bh}(\hat{y}_k, y_k) \tag{5.3}$$

where $b_{\mathcal{S}}$ 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 the Reverse Huber loss $\mathcal{L}_{bh}$ [387] is

$$\mathcal{L}_{bh}(\hat{y}_k, y_k) = \begin{cases} \frac{(\hat{y}_k - y_k)^2 + c^2}{2c} & \text{if } |\hat{y}_k - y_k| > c \\ |\hat{y}_k - y_k| & \text{if } |\hat{y}_k - y_k| \leq c \end{cases} \tag{5.4}$$

where $c = \delta \cdot \max_k |\hat{y}_k - y_k|$ and we set $\delta = 0.05$, which is the value in the code we build upon [333].

In the second step we set $\lambda_{\mathcal{S}} = 1.0$ and $\lambda_{\mathcal{T}} = 1.0$ as we introduce real domain data into the training process using $\mathcal{R}_p$ for supervision. After the filtering step used to obtain $\mathcal{R}_p$ (see Section 5.3.3) a certain number of false positives remains. The noisy points in $\mathcal{R}_p$ are more likely to be further away from the dense depth prediction $\hat{y}$, hence the BerHu loss [387] 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 MAE loss, as MAE weights all values equally, being then less affected by the noise. We thus define $\mathcal{L}_{\mathcal{T}}$ as

$$\mathcal{L}_{\mathcal{T}} = \frac{1}{b_{\mathcal{T}}} \sum_{i} \frac{1}{\#(\mathcal{R}_{p,i})} \sum_{k} |\hat{y}_k - y_k| \tag{5.5}$$

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

5.4 Experiments

We use PyTorch 1.3.1 [246] and a NVIDIA 1080 Ti GPU, as well as the official implementation of FusionNet [333] as our sparse-to-dense architecture. The batch size is set to 4 and we use Adam [162] with a learning rate of 0.001. For the synthetic data, we train by randomly projecting our sparsified depth from the virtual LiDAR camera to the left or right camera with the same probability. In the first step of training, we use only synthetic data (*i.e.*, $\lambda_{\mathcal{S}} = 1.0$, $\lambda_{\mathcal{T}} = 0.0$, $b_{\mathcal{S}} = 4$ and $b_{\mathcal{T}} = 0$) until performance plateaus in the synthetic validation set. In the second step, we mix real and synthetic data setting $\lambda_{\mathcal{S}} = 1.0$, $\lambda_{\mathcal{T}} = 1.0$, $b_{\mathcal{S}} = 2$, $b_{\mathcal{T}} = 2$, the window size of the filter to $w_p = 16$ pixels, the object thickness parameter in the filter to $\theta = 0.5$ m, and train for 40000 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. We will only report results in the outdoor KITTI benchmark as there are no standard real LiDAR+RGB indoor depth completion datasets available. When reporting results in NYUv2 [307], past methods synthetically sparsify the dense ground-truth using a Bernoulli sampling approach, while VOID [343] provides sparse depth from visual inertial odometry that contains no projection artifacts. We thus focus on the KITTI depth completion setting [331], which provides paired real noisy LiDAR depth with RGB images, along with denser depth

ground truth for testing. We use the official KITTI code to evaluate our method in the selected validation set and in the online test set, each containing 1000 images. Following [333], we train using images of 1216×256 by cropping their top part, as they do not contain any input LiDAR depth information. We evaluate on the full resolution images of 1216×356.

Evaluation Metrics. We use the KITTI depth completion benchmark [331] official metrics. The two main metrics are Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), defined as:

$$\text{MAE} = \frac{1}{n_i}\sum_k |\hat{y}_k - y_k| \qquad \text{RMSE} = \sqrt{\frac{1}{n_i}\sum_k (\hat{y}_k - y_k)^2} \tag{5.6}$$

where n_i is the total number of pixels with valid ground-truth for an image i in the test data. The inverse version of these metrics, iRMSE and iMAE, are also widely used. In both iRMSE and iMAE we use the inverse depth $1/\hat{y}$ and $1/y$ in the regular MAE and RMSE expressions. RMSE and MAE give more weight to larger depth values due to those regions having a larger error in absolute terms. However, due to the inverse depth used for their computation, iRMSE and iMAE instead give more weight to the closer depth values, and thus they give complementary information to both RMSE and MAE. In our experiments, following the standard KITTI practice, both RMSE and MAE are reported in mm, and inverse iRMSE and iMAE are reported in 1/km.

Synthetic Data. We employ CARLA 0.84 [78] to generate synthetic data using the camera set-up in Figure 5.1. We collect images from 154 episodes resulting in 18022 multicamera images for training and 3800 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 1392×1392 with a Field Of View of 90°. The dataset is given in `https://github.com/alopezgit/project-adapt`. To match the view and image resolution in KITTI, we first take a center crop of resolution 1216×356, and from this crop we take the upper part of resolution 1216×256. To adapt the synthetic RGB images, we train the original implementation of CycleGAN [382] for 180000 iterations.

Table 5.1: Ablation study on the KITTI 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. *Real GT Supervision* is the original result from [333] trained with a MSE loss, whereas *Real GT + Synthetic GT* refers to our result combining both CARLA and KITTI ground-truth and using a BerHu loss.

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 + Only 1 mask	1273.77	316.22	5.26	1.37
+ 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
$\mathcal{R}_p$+BerHu	1328.76	320.23	4.25	1.33
Full Pipeline: $\mathcal{R}_p$	**1150.27**	**281.94**	**3.84**	**1.20**
Real GT + Synthetic GT	984.85	236.64	2.95	1.10
Real GT Supervision	802.49	214.04	2.24	0.91

5.4.1 Ablation Study

We include an ablation study in Table 5.1 using the validation set. For the result of the whole pipeline, we average the results of three different runs to account for training variability.

CARLA Adaptation. Table 5.1 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 5.1 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 5.4, using image translation further reduces the RMSE by 6.6% when used jointly with real LiDAR masks sampling and projections. Figure 5.5 includes predictions for images 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.

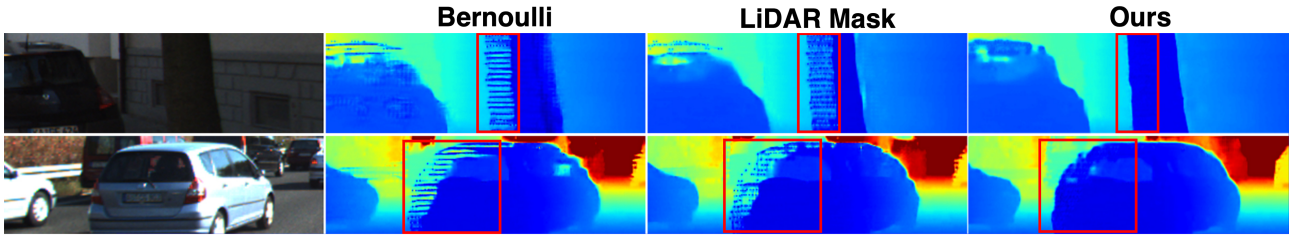


Figure 5.5: Qualitative results with different training methodologies. *Bernoulli* refers to training using the Bernoulli sparsified depth $x_D^{s_B}$, *LiDAR Mask* to training using the depth sparsified with real LiDAR masks $x_D^{s_M}$ and *Ours* to our full pipeline that uses both LiDAR masks and projections. Both rows show projection artifacts, which *Ours* deals with correctly.

Number of Sampled LiDAR Masks. We now aim to test the effect on performance on the number of binary LiDAR masks used, as in real scenarios we may not have access to a large amount of real LiDAR masks. To that end, we sampled a single mask at random from KITTI, which is applied to all the synthetic depth images during training. Table 5.1 shows the performance obtained for our method after the first step of training, *i.e.*, using only synthetic data supervision, when using only one mask (+*Proj.* + *CycleGAN RGB* + *Only 1 mask*), where we see that the performance impact is minimal compared to using all available masks in KITTI (+*Proj.* + *CycleGAN RGB*).

Introducing Real Domain Data. Using the reliable points $\mathcal{R}_p$ as supervision in the real domain alongside the MAE loss function increases the performance as Table 5.1 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 [211, 343] 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 $\mathcal{R}_p$ 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 5.1.

Upper Bound. *Real GT Supervision* in Table 5.1 refers to the results reported in [333], which trains only with KITTI ground-truth and using a MSE loss. *Real GT* + *Synthetic GT* refers to the results when also using KITTI ground-truth but mixing in the batch both CARLA and KITTI data with equal weighting in the loss, and *Berhu* for both KITTI and CARLA data. When having real data

Table 5.2: Results on the KITTI 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
+ T^{2}Net transfer [379]	1184.39	306.66	3.99	1.36
+ FDM [2]	1150.42	**281.11**	3.87	1.21
+ CycleGAN RGB [382]	**1150.27**	281.94	**3.84**	**1.20**

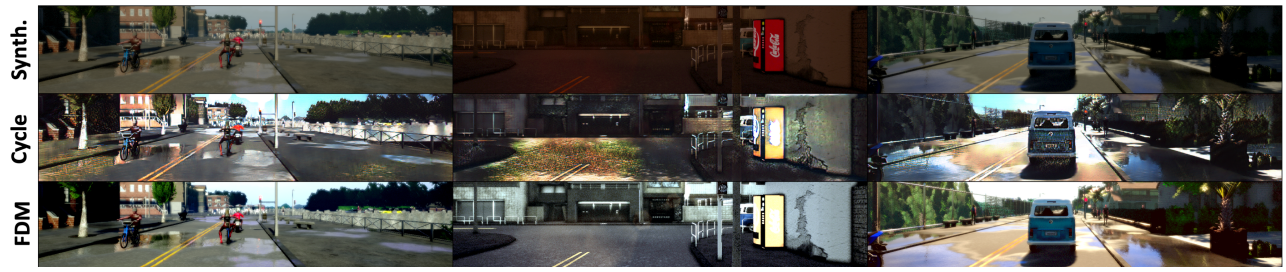


Figure 5.6: Qualitative results when using two different image transfer strategies. We apply to the original CARLA images (first row) either a CycleGAN approach (second row) or a FDM approach (third row).

ground-truth, mixing synthetic and real data affects the attainable upper bound instead of giving a performance increase. However, our method can serve as a good pretraining step in the case where we only have access to a few labeled examples, which will be shown in a later section.

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 5.2. 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%. A similar low impact when adding the RGB modality was found in a past self-supervised work [211]. In a fully supervised manner the difference is 16.3% for FusionNet, showing that methods aiming to further reduce the RGB domain gap may increase the performance. Due to computational constraints, we train the CycleGAN model in a separate step. To test an end-to-end approach, we adapt T^{2}Net [379], which, instead of using cycle-consistency, relies on a depth estimation network to maintain geometric consistency after translation. However, our depth completion network is not capable of properly constraining the translation, and, hence, we obtained lower-quality translated images and larger error compared to CycleGAN as Table 5.2 shows.

Table 5.3: Comparison of results on the KITTI selected validation set and the official online test set. *DA Base* is our Domain Adaptation baseline formed by CycleGAN [382] + LiDAR Masks. RMSE and MAE are reported in mm, and iRMSE and iMAE in 1/km. We compare against the supervised and self-supervised S2D [211] variants and the self-supervised VOICED [343], the unsupervised, self-supervised and supervised variants *DDP* [357], and the supervised FusionNet [333]. The unsupervised *DDP* [357] refers to only using Virtual KITTI data without using any KITTI data. *DA Base* is our Domain Adaptation baseline, with adversarial adaptation with a discriminator at the output (*D. Out*) or in the features (*D. Feat*). *Ours* uses FusionNet architecture, *Ours-S2D* uses S2D architecture and *Ours+SS* uses the video self-supervision from [211]. Bolded results refer to best performance among those methods not using the semi-dense KITTI ground-truth.

		Validation Set				**Online Test Set**			
Model	Param.	RMSE	MAE	iRMSE	iMAE	RMSE	MAE	iRMSE	iMAE
Unsuperv.									
DDP [357]	18.8M	1325.79	355.86	-	-	1285.14	353.16	3.69	1.37
Self-Sup.									
SS-S2D [211]	27.8M	1384.85	358.92	4.32	1.60	1299.85	350.32	4.07	1.57
DDP+St. [357]	18.8M	1310.03	347.17	-	-	1263.19	343.46	3.58	1.32
VOICED [343]	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	2.6M	**1112.83**	**268.79**	**3.27**	**1.12**	**1062.48**	**268.37**	**3.12**	**1.13**
Supervised									
S-S2D [211]	27.8M	878.56	260.90	3.25	1.34	814.73	249.95	2.80	1.21
FusionNet [333]	2.6M	802.49	214.04	2.24	0.91	772.87	215.02	2.19	0.93
DDP [357]	18.8M	-	-	-	-	836.00	205.40	2.12	0.86

Less Demanding RGB Transfer. The translation to real domain performed by CycleGAN on the CARLA images are mostly related to contrast and saturation changes. Figure 5.6 shows that the CycleGAN translation also introduces artifacts similar to the chromatic aberration present in KITTI (*e.g.*, road in the middle column images or the back of the van in the right column example), along with other distortions. Additionally, the results in Table 5.2 show that CycleGAN adaptation does not have a large impact in our final performance. We thus aim to emulate the type of visual changes performed by CycleGAN with a less demanding non-adversarial approach, which does not produce the artifacts introduced by CycleGAN. To that end, we replace the CycleGAN module, which is a standard module used in Domain Adaptation for RGB adaptation, with the Feature Distribution Matching (FDM) ap-

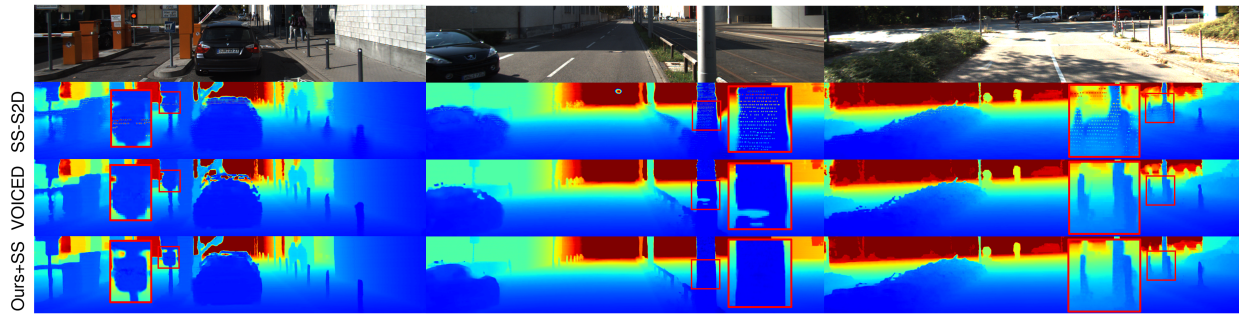


Figure 5.7: Qualitative results on the official KITTI test set for *SS-S2D*, *VOICED* and our method with video supervision *Ours+SS*. Red rectangles show zoomed-in areas.

proach proposed in [2], which does not require training and is less computationally expensive. FDM mostly adapts color and contrast by matching the mean and covariance of the synthetic image to a randomly sampled real image. Figure 5.6 shows examples of the resulting FDM images. In some cases, FDM translates the images to non-realistic colors, but we mostly observe that the translated images are similar to the CycleGAN images but without local artifacts. The performance when training with FDM images is given in Table 5.2, obtaining similar results to CycleGAN.

5.4.2 Method Evaluation

Comparison to State-of-the-Art. In Table 5.3 we compare our method, *Ours*, with the state-of-the-art methods not using real domain ground-truth. Our method improves upon the GT-free state-of-the-art methods, and specifically decreases in the test set the RMSE by 6.4%, the MAE by 6.3% and obtains better results for iRMSE and iMAE compared to VOICED [343], which uses video self-supervision. These improvements are obtained using an architecture with fewer parameters, and without using any self-supervision. Table 5.1 and Table 5.3 show that we achieve similar results to VOICED [343] 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 [355] uses synthetic ground truth from Virtual KITTI [92] for training, however no adaptation is performed on the synthetic data, resulting in worse results compared to *Ours* even when using stereo pairs (*DDP+St.*). Both VOICED [343] and SS-S2D [211] use the unfiltered noisy sparse input as supervision, reducing the achievable performance as shown in Table 5.1 in *No Filter*. Regarding fully-supervised methods,

Table 5.3 shows that we obtain a MAE of 214.04 mm whereas the fully-supervised FusionNet, which is our base architecture, obtains a MAE of 268.79 mm, *i.e.*, our method has 25% more error than the fully-supervised approach.

Adding Self-Supervision. When real domain video data is available, our approach can be combined with self-supervised methods [211, 343]. We apply to our pipeline the approach from [211], where the pose between two video frames is computed by combining the input LiDAR depth points with 2D matches computed using SIFT features, and that pose is used to compute a multi-scale photometric loss $\lambda_{ph}\mathcal{L}_{ph}$. We add the photometric loss $\lambda_{ph}\mathcal{L}_{ph}$ during the second step of training only for the real KITTI data, where we set $\lambda_{ph} = 10$ to have similar loss values as $\mathcal{L}_S + \mathcal{L}_R$. *Ours+SS* in Table 5.3 further reduces the error in the test set and achieves, compared to VOICED [343], a lower RMSE by 9.2% and a lower MAE by 10.4%. We also show qualitative results of *Ours+SS*, VOICED and SS-S2D in the official KITTI test set in Figure 5.7. We see that the *SS-S2D* results include sparse see-through artifacts, *e.g.*, the light pole in the middle column images, as SS-S2D uses the noisy depth input as supervision. VOICED handles better these artifacts due to the scaffolding and refinement approach used, but the see-through depth points still affect the results as shown in the artifact present also in the light pole in the middle column images. Our results show a reduction of these see-through artifacts, and additionally better object boundaries compared to either *SS-S2D* and *VOICED*, where edges tend to be not straight and wider than the original object.

Domain Adaptation Baselines. Following synthetic-to-real depth estimation methods [10, 379, 209], we use as a Domain Adaptation baseline an image translation method, in our case CycleGAN [382], to adapt the synthetic RGB images. To sparsify the synthetic depth, we use the real LiDAR masks [11], shown in Table 5.1 to perform better than Bernoulli sampling. The performance of this Domain Adaptation baseline is presented in Table 5.3 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 5.3 uses an output discriminator using the architecture in [253], with an adversarial loss weight of 0.001 similarly to [328]. Following [379], we also tested a feature discriminator in the model bottleneck in *DA Base + D. Feat.* with weight 0.01. Table 5.3 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.

Table 5.4: Semi-supervised results on the KITTI 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**

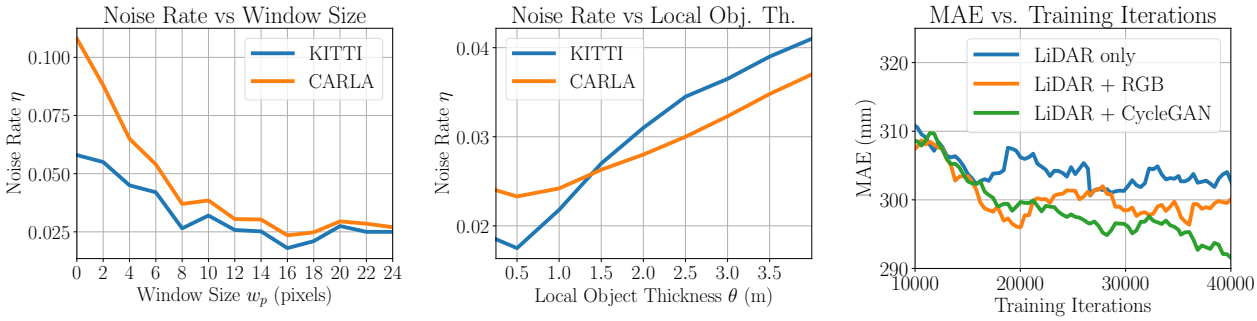


Figure 5.8: Hyperparameter analysis. The two left images show the noise rate η vs. w_p ($\theta = 0.5$ m) and θ ($w_p = 16$ pixels). The right plot shows MAE vs. training iterations in the 2nd step, evaluating every 400 iterations, and using a moving average of 25 values with further averaging of 3 runs.

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 $\mathcal{R}_p$ for supervision. This provides a good initialization for further finetuning with any available annotations as Table 5.4 shows. Compared to pretraining using the *DA Baseline*, our method achieves in all cases a better performance after finetuning.

Hyper-Parameter Selection. We do not tune the loss weights $\lambda_\mathcal{S}$ and $\lambda_\mathcal{T}$. 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 $\mathcal{R}_p$ as the indicator for the filtering process performance. Figure 5.8 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 5.8 also shows the MAE depending on the number of iterations in the second step. After 40000 training iterations, we did not see any improvement.

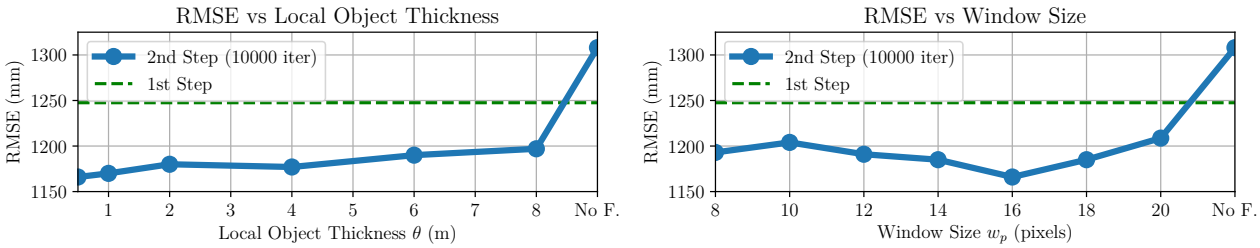


Figure 5.9: RMSE in the second step of training when varying the parameters of the filter used to obtain the reliable points $\mathcal{R}_p$. In the *left image* we vary the object thickness θ with window size set to $w_p = 16$ pixels. In the *right image* we vary the window size w_p, with object thickness fixed to $\theta = 0.5$ m. *1st Step* is the performance obtained after the first step of training, *i.e.*, only using synthetic data. *No F.* refers to not using any filter in the real sparse depth used for supervision during the second step of training.

Sensitivity to Filter Parameters. Figure 5.9 shows the RMSE when varying the window size w_p and the object thickness θ after 10000 iterations of second step of training. For parameters different to our base values of $\theta = 0.5$ m and $w_p = 16$ pixels, the performance after 40000 iterations is decreased compared to the performance after 10000 iterations, and also lower than the performance after only the first fully-synthetic step of training. We hypothesize that the reason for the drop of performance after 40000 iterations for non-optimal filter parameters is the increased noise rate η. Hence, a smaller number of training iterations is a safer choice for a wider range of filter parameters to avoid overfitting to the noisy points remaining after the filtering step.

Model Agnosticism. We chose FusionNet [333] as our main architecture, but we now test our approach with the model used in [211] to show our method is robust to architectural changes. Due to memory constraints we use the 18-layers architecture instead of the 34-layers model from [211], which accounts for the different parameter count in Table 5.3 between *Ours-S2D* and *SS-S2D*. We set the batch size to 2, increase the iterations in the second step to 90000 (the last 20000 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 5.3 in *Ours w/S2D arch.*, which achieves state-of-the-art RMSE and MAE.

Qualitative Results on PandaSet [299] are shown in Figure 5.10 for our method and the DA Baseline, both without further tuning in PandaSet. PandaSet contains a different camera set-up and physical distances compared to the CARLA/KITTI training set-up, *e.g.*, top two rows in Figure 5.10 correspond to a back camera not present in KITTI, whereas the two bottom rows are from a frontal-left

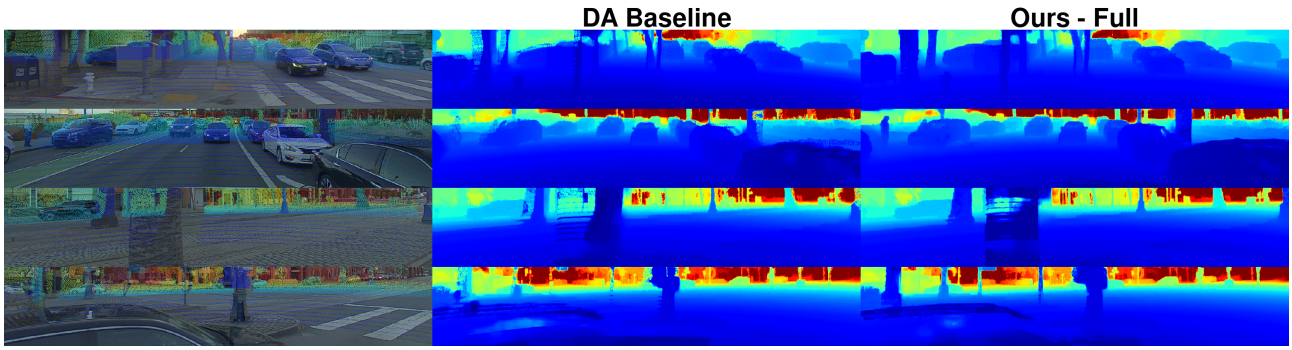


Figure 5.10: Qualitative results on PandaSet [299] for our *DA Baseline* and full method (*Ours - Full*) trained in CARLA and KITTI. RGB images also show overlaid the sparse depth input.

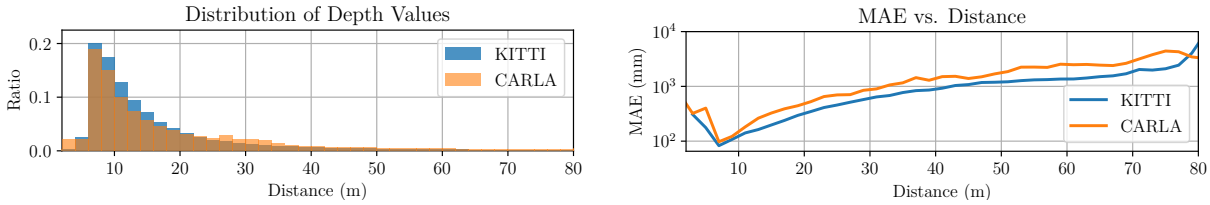


Figure 5.11: Left: Distribution of depth ground-truth values in the KITTI validation set and the CARLA training set. Right: MAE vs. ground-truth depth values in the KITTI validation set and the CARLA training set for our final model.

camera. Our method is still capable of dealing better with the projection artifacts and with the missing data compared to the DA Baseline. However, there are regions that our method is not capable of correcting completely, such as the see-through artifacts in the tree trunk in the third row, which is due to a different camera set-up and depth statistics compared to our CARLA dataset. PandaSet does not provide depth completion ground-truth, thus no quantitative results can be computed.

Distribution of Errors. Figure 5.11 shows a similar distribution of ground-truth depth values in CARLA and KITTI, as the synthetic data matches the scenes, camera position, intrinsics and crop used in KITTI. Figure 5.11 also shows that, for both real and synthetic data, the MAE of the predictions increases for larger depth values, having both datasets have a minimum MAE for values around 6-8 m, where most of the depth values are concentrated. The error in the CARLA training set is consistently higher for most depth ranges, obtaining a total MAE of 500 mm, compared to the 282 mm obtained in the KITTI validation set. This higher error in the CARLA training data is a result of the synthetic ground-truth being completely dense, whereas, due to the process used to generate the semi-dense maps, the KITTI ground-truth has a bias towards easier to predict depth values. To

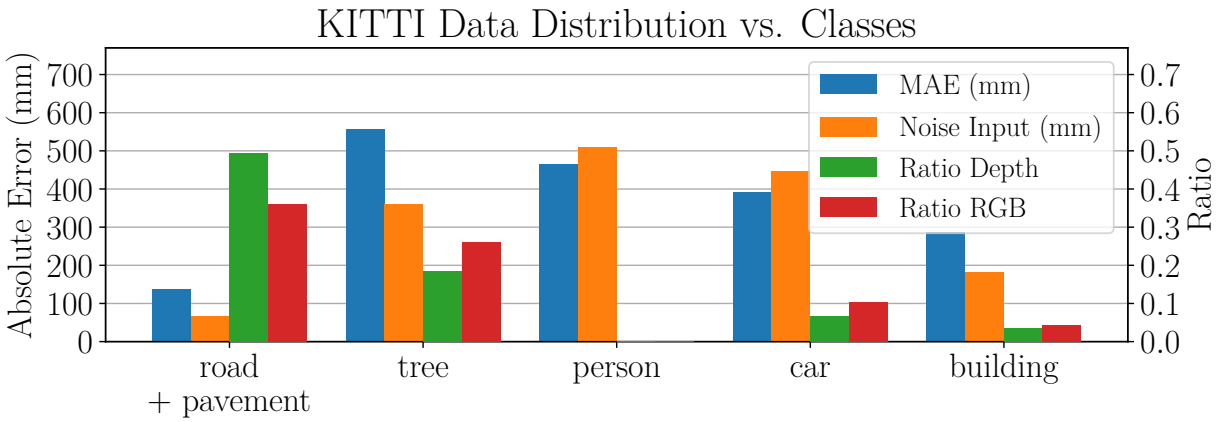


Figure 5.12: Error and data distribution depending on the detected (with Mask R-CNN [121] pre-trained on COCO [345]) class on the KITTI selected validation set for the crop of 1216×256. *MAE* is the mean absolute error, *Noise Input* is the mean absolute error in the input depth values, *Ratio Depth* is the ratio of the depth GT values available divided by the total amount of ground-truth values, *Ratio RGB* is the ratio of pixels representing a class divided by the total amount of pixels.

show this bias, in Figure 5.12 we plot for different semantic classes in KITTI the MAE, the average input noise and the ratio that each class represents in the available ground-truth depth (*Ratio Depth*) and the RGB images (*Ratio RGB*). Figure 5.12 shows a relationship between the input noise and the prediction MAE, where classes with high input noise, *e.g.*, *tree*, also present high output MAE. In KITTI, the class *road+pavement*, which has both the lowest output MAE and input noise, is given more weight in the depth ground-truth (*Ratio Depth*) compared to the actual ratio of RGB pixels that are from that class (*Ratio RGB*). The input noise of the different classes is related to the see-through artifacts present in the input LiDAR depth, which is the reason why *road+pavement* (where there are no see-through artifacts) presents a low input noise. Furthermore, Figure 5.13 shows that the KITTI ground-truth has missing regions around the instances edges, which are usually areas of high prediction error. These missing regions contribute to the lower error in the KITTI validation set compared to the CARLA training set, and are a consequence of the consistency step performed in the KITTI depth completion ground-truth [331]. This consistency step is performed between the accumulation of 11 consecutive LiDAR depth frames and a stereo-based Semi-Global Matching (SGM) method to verify the correctness of the accumulated LiDAR depth. Due to the bleeding edge effect [98], SGM shows missing regions around the edges of the objects, and thus those regions are also removed from the final clean accumulated LiDAR depths as the correctness of those areas cannot be verified.

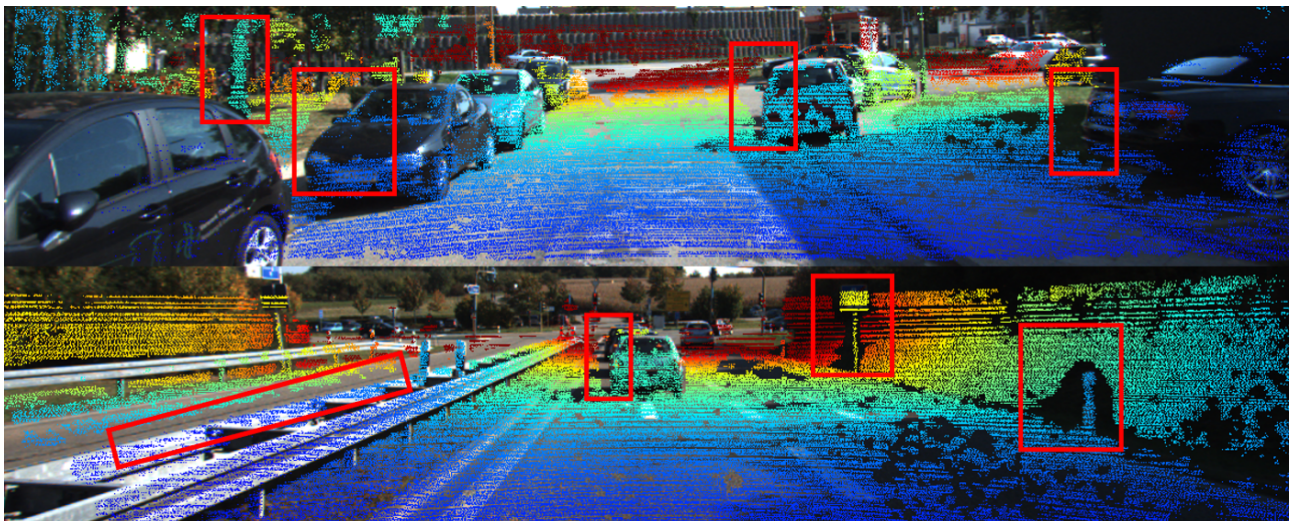


Figure 5.13: KITTI ground-truth overlaid on the RGB image. The red rectangles show challenging areas with missing ground-truth due to the ground-truth acquisition process used in [331].

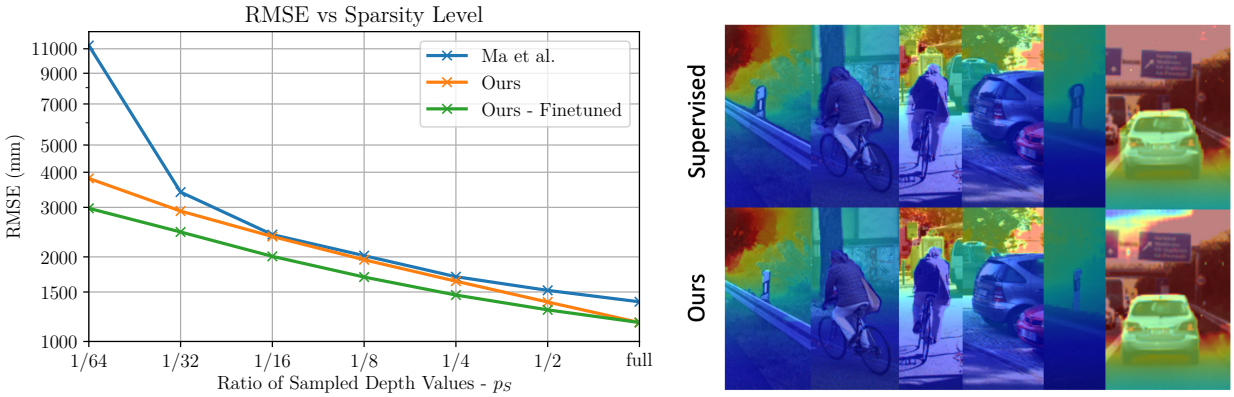


Figure 5.14: Left: RMSE obtained when varying the sparsity level by sampling the input sparse depth with probability p_S. Right: Methods supervised with KITTI ground-truth (top) expand the predictions around the edges compared to our method (bottom), which uses synthetic supervision.

Sparsity of Input Data. Similarly to [211], we evaluate the performance of our method when varying the input sparsity level. In this experiment, we keep each of the given input KITTI sparse depth with probability p_S. We evaluate both our final model trained with the original LiDAR maps, and also our final model finetuned for the different sparsity levels for 10000 iterations with a learning rate of 10^{-4}. Figure 5.14 shows the results of the experiment when varying p_S. We show a similar trend to the results in [211], however our method is capable of performing better over all ranges of values. Finetuning the model increases the accuracy, but even without any further training the model behaves well for lower values of p_S.



Figure 5.15: 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 a dropping of points due to low-reflectance black surfaces.

Ground-Truth Issues. The process performed in [331] to obtain the KITTI semi-dense ground-truth is not completely noise-free, especially in dynamic or far-away objects. Similarly, the ground-truth is not perfectly aligned around the edges of some objects due to small calibration issues in the depth accumulation process. This misalignment of the ground-truth depth coupled with the already mentioned missing depth around some instances shown in Figure 5.13, drives the models trained with real KITTI ground-truth to predict depth maps that expand the instances borders as shown in Figure 5.14. Conversely, our model shows predictions better aligned around the boundaries (*e.g.*, the car on the rightmost column shows a *halo* around the instance borders for the *Supervised* case compared to *Ours*), which highlights the benefits of training using pixel-perfect synthetic depth data.

Computational Complexity. Our method does not add any complexity, thus the inference speed depends on the architecture used. In our work, we use FusionNet, which has a low number of parameters than most of the state-of-art models as shown in Table 5.3, and has a low computational time [130]. When using a 1080 Ti GPU and a batch of 10 images with a resolution of 1216×256, FusionNet is achieves a speed of 35 frames per second. As for training, with our hardware setup it takes ≈ 5.5 hours for the first fully synthetic training step, and ≈ 4.5 hours for the second training step, amounting to a total training time of ≈ 10 hours. This training time does not include the CycleGAN step, however as shown in Table 5.2 using the original RGB images or the lower-complexity FDM [2] method, which does not need any training, obtains a similar performance to using the CycleGAN images.

5.4.3 Limitations

While we addressed see-through artifacts, other types of noise can be present in the real sparse depth as Figure 5.15 shows. The upper row example shows a set of noisy inputs on the wall that is not

corrected when using our approach. Similarly, the bottom row 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. These examples suggest that approaches focused on other types of noise could further decrease the error. Additionally, our method mimicked the KITTI set-up, *e.g.*, camera positions and image resolution used. The performance of our method needs to be assessed quantitatively in diverse set-ups. However, currently there is a lack of real LiDAR depth completion datasets that provide a ground-truth.

5.5 Conclusions

We provided a hypothesis for the reduced performance of a depth completion pipeline in the real domain when naively training with synthetic data. Whereas past methods sampled the input sparse depth directly from the ground-truth, we hypothesise that adapting the input synthetic depth to be similar to a real input LiDAR map, which contains noise, is crucial, whereas RGB adaptation is secondary. We based our method on the assumption that the main source of noise in a joint RGB + LiDAR set-up is the see-through artifacts due to projection from the LiDAR to the RGB reference frame. Based on this assumption, 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. Furthermore, compared to past methods that directly use the input real sparse depth as supervision, we showed that guiding the output with the raw LiDAR input can harm the performance of the model. On that note, we proposed a method to find 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. Overall, our method shows better performance than all the past methods that do not use real domain ground-truth. We also showed that our method can be combined with past self-supervised approaches for improved performance. As we focused on the main source of error, *i.e.*, the see-through artifacts, our method does not correct all the sources of noise in the input data. 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.

Chapter 6

Improving Camera Pose Regression via Synthetic Views

In this chapter, we focus on the synthetic data generation pipeline for a camera pose estimation task, specifically for regression methods (see Section 2.5 for pose estimation background). We motivate our approach by showing that the differences in train and test camera pose distributions affect the performance of the pose regression models. Hence, we propose a sampling strategy for generating synthetic data to tackle that distribution mismatch. Our method reduces the gap in performance between regression methods and the more accurate 3D-based methods.

6.1 Introduction

Visual localization (also known as camera pose estimation) is the problem of localizing an image with respect to a known and pre-mapped environment. This could be in the form of coarse place recognition [8, 326] or predicting the accurate 6DoF camera pose [154, 293] that includes both rotational and positional estimations. The latter is especially important for enabling mass deployment of technologies in robotics, autonomous driving and spatial AR.

Traditionally, localization methods used local features [81, 210, 267, 324], 3D point clouds computed using Structure from Motion [290, 292, 302], and minimal solvers [94] with RANSAC [88] for pose

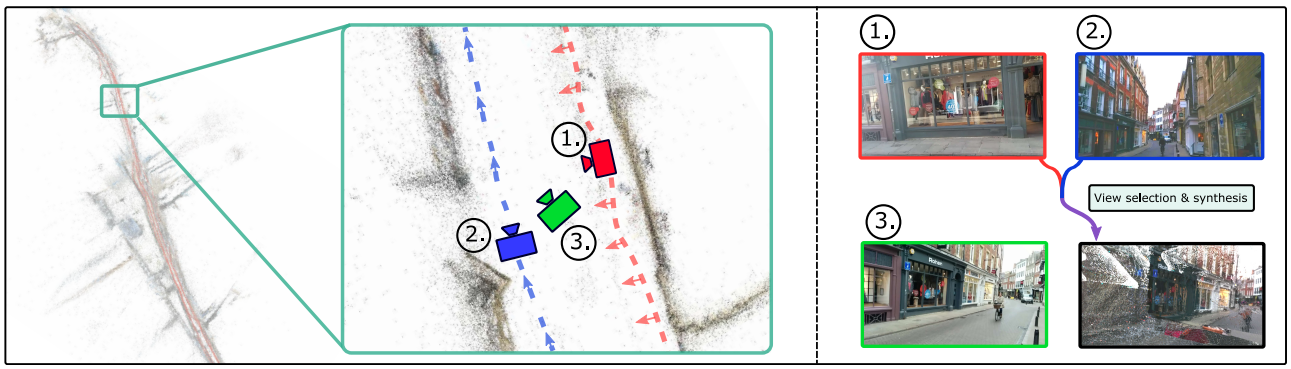


Figure 6.1: Challenge facing CNN methods for camera pose regression. We show a bird's-eye view of the 3D model of a street. The (1) and (2) dotted lines and arrows are the trajectories and orientations of two separate training sequences. As the training data is heavily biased towards specific views, CNN methods do not extrapolate well to the test pose in (3). Our pose regression method alleviates this by learning from (4) synthesized views sampled from an unbiased pose distribution.

computation. These methods have up to now dominated the relevant benchmarks, and are able to predict highly accurate 6DoF pose [293, 294] but are costly in both time and memory. With the advent of CNNs, PoseNet [154] demonstrated that it is possible to directly predict 6DoF camera position in a known scene by end-to-end learning with a convolutional network. Many pose regression works soon followed suit [17, 75, 153, 283, 338] due to the appeal of fast inference, no need for 3D models during inference, and increased potential of deployment in different devices compared to traditional 3D methods.

Despite the initial success, pose regression methods have been on the back foot since the study by Sattler *et al.* [296] showed that the performance of pose regression methods is closer to less accurate image retrieval [254] than to 3D structure-based methods [292]. This is due to the fact that learning-based methods do not extrapolate well beyond the poses they encounter in training. This extrapolation issue was shown in the analysis in Sattler *et al.* [296], where it is explained that absolute pose regression methods predict the final pose weighting a set of base poses from the training set. This problem is especially important in the context of visual localization [103, 154, 293], since the test data is often sampled during a rather structured acquisition process, *e.g.*, a pedestrian walking down a road in a straight continuous trajectory [154] or a driving car [212] showing little variations in camera orientation. This particular bias is illustrated in Figure 6.1 where we observe that in a scene where existing learning-based methods fail to learn or localize (*Street* [154]), the camera poses are biased towards particular viewpoints and orientations. Although prior works attribute this to the incapacity of the CNN to learn large and complex scene structures [28, 296], we argue that the network is capable of

achieving high accuracy if unbiased data is available. One way to address this issue would be to increase the size of the training data with random views. However, this is costly and does not guarantee to remove the bias. Instead we focus on generating synthetic images with a probability-guided process with the aim to reduce the gap between training and test distributions. Compared to the past works presented in this thesis, in this work we do not tackle a standard Unsupervised Domain Adaptation approach, and instead we focus on using synthetic data to improve the performance of our pipeline.

Our main contributions are in the context of the observations by Sattler *et al.* [296], which can be summarized by the following quote:

> *"Pose regression techniques are often closer in performance to image retrieval than to structure-based methods.... Even with large amounts of training data, pose regression does not reach the same performance as structure-based methods."*

Specifically:

- We propose to use synthetic data to reduce the gap between structure-based and regression methods, showing that regression networks may compete with 3D methods with less overhead. We motivate our use of synthetic data by the biased distribution of poses present in some scenes, which limits the performance of a relative pose regression method when sampling training pairs from that distribution.
- Our approach is the first regression method to improve upon image retrieval on the largest and most complex *Street* scene from [154].
- We propose a pipeline for training camera pose regression methods based on a probability-driven selection and generation of synthetic views. The generation of synthetic views is done using multi-image projection and a combination of sparse but up-to-scale ground-truth depths with densely predicted scale-agnostic depths to improve the quality of synthetic images.
- We show that a global regression network is robust to the image artifacts, being able to boost the results of the pipeline with synthetic views of variable quality.

- We show that a Transformer [334] based architecture is more suitable for inferring relative pose regression, outperforming the standard MLP approaches adopted by all previous methods.

6.2 Related Work

We now present the relevant literature for structure-based methods, and also for our view-synthesis proposed method, which we use to bridge the gap between pose regression and 3D-methods (for pose regression background and Domain Adaptation related work see Section 2.5). We also introduce a brief review of Transformers, as we build our regression head using a Transformer-based model.

6.2.1 View Synthesis

There are two main categories of methods for novel view synthesis. The first is based on surface modelling, which explicitly exploits the geometry, *e.g.*, with depth maps or surface meshes [60, 240], to generate an image by projecting the 3D structure onto the novel view. They generalize to a variety of views, and while normally suffer from low levels of photo-realism, blending techniques can greatly improve the quality of images [270].

The second category is volumetric and rendering based, which densely represent the appearance of the scene implicitly with techniques such as multi-plane images [312] or Neural Radiance Fields (NeRF) [224]. They produce highly photo-realistic images, but do not generalize to large viewpoint changes. Recent improvements on modelling uncertainties [220] made it possible to generalize to challenging scenes. Nevertheless, volumetric methods require significant amount of training, are more complex and much slower than surface modelling methods despite recent efforts to improve the speed of convergence [198].

For visual localization, the use of synthetic views has been explored in place recognition [326], pose verification [321, 373] and pose estimation [232, 296, 339]. Unlike the naive data augmentations in [27, 186, 187] which include only in-plane transformations, [232, 339] leverage 3D using a depth prediction network. However, this limits the range of transformations possible around existing poses

for absolute pose regression, making it closer to the data augmentation approach. As part of the study, Sattler *et al.* [296] investigated grid-based pose densification using Multi-View Stereo (MVS) in a small-scale experiment on a single scene, showing the potential of view synthesis in improving both image retrieval and absolute pose regression.

6.2.2 Transformers for Visual Localization

The Transformer [334] has been broadly adopted by the vision community in various tasks, but its performance still needs to be further investigated. The self-attention mechanism has been recently shown to be beneficial for image retrieval [236] and absolute pose regression when applied to feature maps [340]. SuperGlue [291] applies self and cross-attentions for better feature matching in a structure-based approach, achieving state-of-the-art in several localization benchmarks [147, 293].

With attention being the central component of the Transformer, these results are promising for the potential of Transformers in relative pose regression. This motivates the examination of its architecture in the context of this work.

6.3 Method

In this section, we present the main components of our novel localization approach for a pinhole camera model. First, we discuss how we select and generate synthetic views. We then show how to make use of these synthetic images for training a Transformer-based relative pose regression model.

6.3.1 Preliminaries

Projecting to a Novel View with Depth Map. Given that an RGB image $\mathcal{I} \in \mathbb{R}^{h,w,3}$ and its depth map $\mathcal{D} \in \mathbb{R}^{h,w}$ exist in a known 3D environment, *i.e.*, we have both the camera intrinsic $\mathbf{K}$ and extrinsic $\mathbf{C}$ matrices from 3D reconstruction, we are able to obtain the synthetic image from a novel view $\hat{\mathcal{I}}$ with three steps. **1.)** For a pixel $\mathbf{p} = (u, v)^\top$ in $\mathcal{I}$, its 3D point in world coordinate $\mathbf{x} =$

$(x,\, y,\, z)^\top$ is obtained by $\mathbf{x}(\mathbf{p}) = \mathbf{C}\mathcal{D}(\mathbf{p})\mathbf{K}^{-1}\tilde{\mathbf{p}}$, where $\tilde{\mathbf{p}} = (u,\, v,\, 1)^\top$ denotes the homogeneous vector. **2.)** We denote the extrinsic matrix of the novel view $\hat{\mathbf{C}} = T(\mathbf{C})$, where $T \in \mathbb{SE}(3)$ is the transformation from the source view $\mathbf{C}$ to our target novel view $\hat{\mathbf{C}}$. **3.)** Lastly, we re-project the 3D point $\mathbf{x}$ to image plane of $\hat{\mathcal{I}}$ by $\hat{\mathbf{p}} = \pi(\hat{\mathbf{x}})$, where $\hat{\mathbf{x}} = (\hat{x},\, \hat{y},\, \hat{z})^\top = \mathbf{K}\hat{\mathbf{C}}^{-1}\mathbf{x}$, and π denotes the dehomogenization $\pi(\hat{\mathbf{x}}) = (\lfloor \hat{x}/\hat{z} \rfloor,\, \lfloor \hat{y}/\hat{z} \rfloor)^\top$. We compact the function that returns the new depth value(s) $\hat{z}$ at this point as $\boldsymbol{\xi}_{proj}(\mathbf{p})|_{\mathcal{D},T} := \{\hat{z} \in \hat{\mathbf{X}}_z \,|\, \pi(\hat{\mathbf{X}}) = \hat{\mathbf{p}}\}$, the target image $\hat{\mathcal{I}}$ is hence formed by retrieving all corresponding RGB values from $\mathcal{I}$,

$$\hat{\mathcal{I}}(\hat{\mathbf{p}}) = \begin{cases} \mathcal{I}\left(\arg\min_{\mathbf{p}} \boldsymbol{\xi}_{proj}\left(\mathbf{p}\right)|_{\mathcal{D},T}\right), & |\hat{z}| > 0 \\ (1,1,1)^\top, & \hat{z} = \emptyset. \end{cases} \tag{6.1}$$

The first condition in Equation (6.1) ensures that occlusions are retained, and the second condition that holes from warping the depth maps are filled with a white background.

Motivation - the Infamous *Street* Scene. Recently, Sattler *et al.* [296] demonstrated that absolute pose regression methods perform closer to image retrieval than 3D-based methods. It is also noted that no regression model so far has been able to produce any meaningful result on the largest scene in Cambridge Landmarks [154] - the *Street* scene. In fact, the network used in DSAC++ [28] for scene coordinates regression, which is state-of-the-art in all other scenes, does not converge in *Street*. Also, no current regression method, either absolute or relative, consistently improves upon image retrieval on this scene. Although some previous works have speculated the size of the scene and complexity being the main reason for this [28, 296], these claims are not backed-up experimentally. In the following, we show that this view is not correct, and the culprit is a biased distribution of training poses instead.

This is illustrated in Figure 6.2, where we compare the distributions of camera angles about the *yaw* axis (rotating 'left-to-right' from a standing person perspective) between the training sets of the *King's College* and the *Street* scenes in Cambridge Landmarks [154] dataset. We observe that while the rotations in *King's College* form a flat distribution, there may be a bias within *Street* training distribution showing distinct multiple modes with no angles in between. There are four peaks at

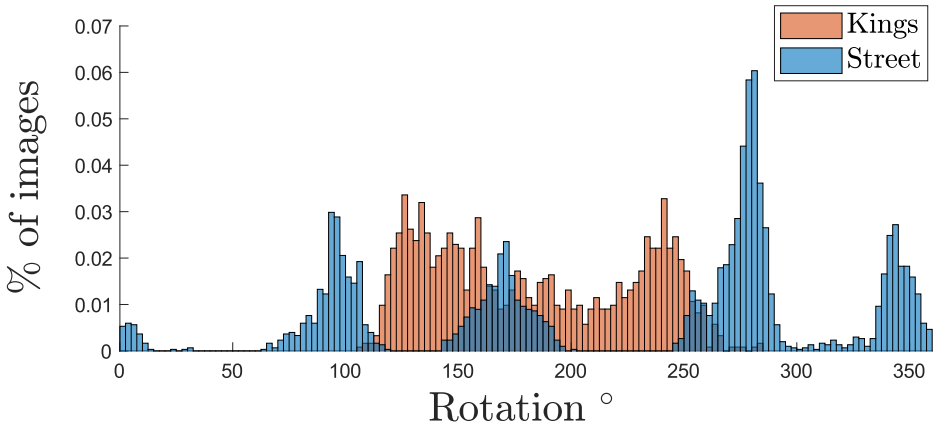


Figure 6.2: Distribution of absolute *yaw* rotations of *Kings* vs. *Street* in the Cambridge Landmarks [154] dataset. The four peaks in *Street* show there may exist large bias in the data collection process compared to the more uniform *Kings* distribution.

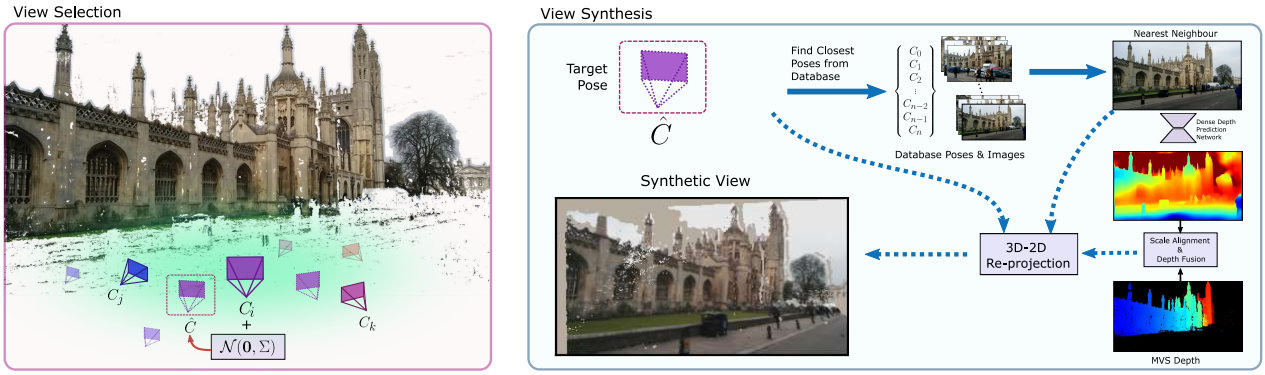


Figure 6.3: Illustration of view selection and synthesis for training. Left: We first sample a target pose $\hat{C}$ by a Gaussian (opaque green cloud) based on an existing pose C_i. Right: We produce synthetic images from $\hat{C}$ by combining densely-predicted but scale-agnostic depths with the sparse metric MVS depths of the nearest images, and re-projecting to the target pose.

90° intervals, which suggests that the training images come from four sequences, each facing one predominant direction. Intuitively, for a camera pose, such data may violate the inductive bias that allows regression methods to work in other scenes, which is that the training distribution should approximate the test distribution.

Without access to the test data, our best assumption is that the test poses should be uniformly distributed. Hence, the bias problem becomes apparent, since in both absolute and relative pose training for *Street*, the network only learns from poses in 90° intervals. Therefore, we cannot expect the network to be able to extrapolate far beyond such a narrow range of training poses.

To further experimentally address the question *'Is Street too large and too complex for a network to learn, or is it learnable if we fix the bias in the data?'*, we conduct a simple sanity check. We first use

Table 6.1: Sanity check on *Street*. For the first and second experiments we train on test views with real images and synthetic views generated with training images respectively. The last row is the coarse prediction with image retrieval.

Experiment	Median Error
Real views from test poses (Upper-bound)	0.89 m, 1.50 °
Synthetic views projected onto test poses	3.54 m, 13.9 °
Image retrieval with GeM [260]	6.42 m, 23.6°

image retrieval to find the nearest neighbor in the training set for each test query. We then overfit the relative poses of these test image pairs to our *Pose Regression Module* (see Section 6.3.3 for in-depth explanation of the module) to serve as an indicator of the upper-bound accuracy.

With the *Views Selection* and *Synthesis* modules from Section 6.3.2, we produce synthetic views from the training images projected onto the test poses and train the *Pose Regression Module* with these synthetic views and their nearest neighbor. Finally, we predict the poses for the test images and their nearest neighbor in training data.

The results are presented in Table 6.1. Given that the upper-bound results reduce the median errors from retrieval nearly ten-fold, is it evident that the scale/complexity of the scene is not the main issue. Training with known synthetic poses also reduces the error by half compared to the coarse prediction by retrieval. This suggests that the network is able to learn on synthetic views and improve its generalization from training data, even in large-scale scenes like *Street* which covers more than 50000 m^2 [154].

6.3.2 View Selection and Synthesis

Given our hypothesis that the distribution of training poses is a potential factor for improving pose regression, we introduce in this section a novel way of acquiring synthetic views for the densification of the training distribution/alignment with test distribution. There are two components - *View Selection* and *View Synthesis*, which are illustrated in Figure 6.3.

View Selection. To generate our synthetic views, we select camera poses based on a multi-variate

Gaussian distribution $\mathcal{N}(\mathbf{0}, \Sigma)$ around the existing poses $\{C_i, C_j, C_k, ...\}$. Using Gaussian increases the probability that the new pose will be close to the existing one thus have sufficient overlap with real images needed for synthesis. The Gaussian density is visualized by the opaque green cloud on the left of Figure 6.3. The goal of this component is to select a target pose $\hat{C}$ that alleviates the bias in the training data demonstrated earlier. This is illustrated in Figure 6.3 left again, where $\hat{C}$ is ideally beyond the interpolation C_i and C_j to achieve the filling of missing poses in the overall distribution. We propose two variants of this selection process based on our assumptions on the data: *In-Distribution* and *Out-of-Distribution*.

In-Distribution: We assume that the inductive bias holds, *i.e.*, the training distribution approximates the test distribution. For each training image, we obtain a set of nearest neighbors (NNs) using a retrieval method, in our case GeM [259], which we use to form our training query-NN pairs. The target pose $\hat{C} = \left(\hat{q}, \hat{t}\right)^\top \in \mathbb{R}^7$ is obtained using the pose of the nearest neighbor C_{NN} as a basis

$$\hat{C} = C_{NN} + \mathcal{N}\left(\mathbf{0},\ \Sigma_{in}\right) \tag{6.2}$$

where $\Sigma_{in} = \begin{bmatrix} \alpha_q I_4 & \mathbf{0} \\ \mathbf{0} & \alpha_t I_3 \end{bmatrix}$ and α_q, α_t are scalar multipliers for the quaternion and translation vectors that aim to add small scale perturbations around C_{NN}. After the perturbation, the rotation quaternion is normalized to 1.

Out-of-Distribution: We assume that a large bias might be present such as in the *Street* example from Figure 6.2. In this case, we cannot base our synthesized views only on the existing training poses as the network cannot extrapolate out of those ranges. Instead, we generate new poses by taking each training image and applying to its pose a large perturbation compared to *In-Distribution*. This perturbation is parametrized by the Gaussian $\mathcal{N}(\mathbf{0},\ \Sigma_{out})$ around the training query poses, where Σ_{out} is determined from the training data. For example, for outdoor scenes such as Cambridge [154], we observe that most images are viewed from roughly the same height and the camera elevation does not vary as much compared to movements on the ground plane. Therefore, the scale of perturbations along the two ground plane axes would be larger than the vertical. For camera translations, this might result in poses where no 3D-points are visible, *e.g.*, the camera centre might end up inside a wall. On

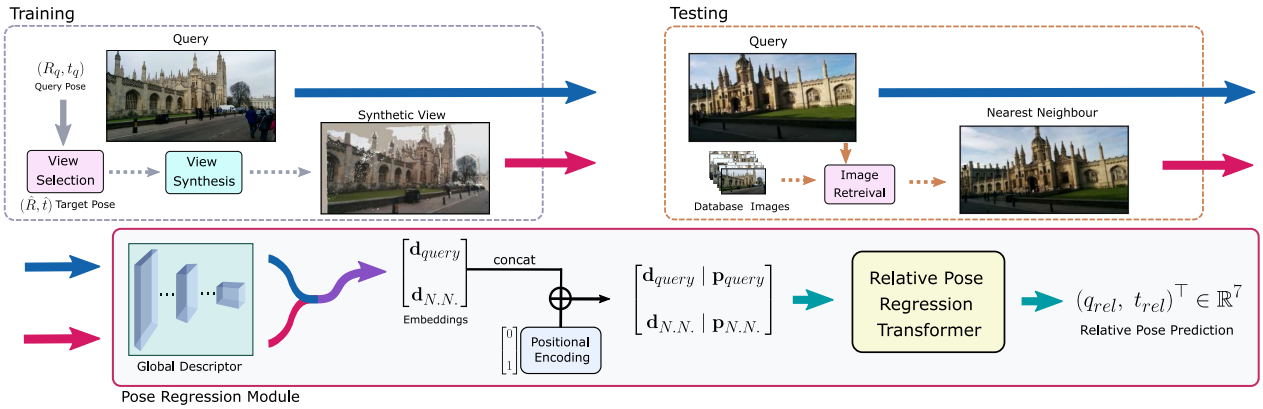


Figure 6.4: Our localization pipeline. The image pair - query and nearest neighbor (N.N.) - are represented by blue and red arrows. During training, the N.N. is synthesized from the process described in Section 6.3.2. In inference, N.N. is determined by an image retrieval method. The image pair is fed into our *Pose Regression Module*, which contains a Transformer that predicts the relative pose between the two.

that note, we check within a spatial neighborhood if there are existing camera poses and use that with an additional small perturbation, as the target translation $\hat{t}$. For the target rotation $\hat{q}$, we keep what is selected from adding $\mathcal{N}(\mathbf{0},\ \Sigma_{out})$ to the query rotation.

View Synthesis. This is illustrated on the right of Figure 6.3. For each of the training images, we obtain a depth map from a Multi-View Stereo process. The resulting MVS depth can be sparse, which affects the quality of our synthetic images. We therefore fuse the MVS depth map with a densely predicted depth using MiDaS [262]. Since the MiDaS predicted depth maps do not have a consistent scale, we align its scale with ground-truth sparse MVS depths by multiplying it with the ratio of their medians, as used in past depth estimation evaluation protocols [381]. We then fill the missing depths (black regions in the 'MVS Depth' photo in Figure 6.3) with the now scale-aligned dense depth prediction. For the *In-Distribution* approach, we expect our target poses to be close to the poses in the training pairs. Hence, we directly project the NN to the new pose using the fused depth with the process described in Section 6.3.1. However, in *Out-of-Distribution*, the new synthetic pose can be distant to the training pose it is generated from. We therefore iteratively re-project 3D points according to the pose distance of the source image to the target pose, *i.e.*, the vector $(1, 1, 1)^\top$ in the second condition of Equation (6.1) is replaced by the corresponding RGB values from the projection process of the next closest image. This is repeated until the image is filled to a certain threshold or a maximum number of iterations.

6.3.3 Localization Pipeline

Our full visual localization pipeline is illustrated in Figure 6.4. The *Pose Regression Module* takes two images, including the query (blue arrows) and NN (red arrows), as input in both training and testing. In training, we sample synthetic views using the process described in Section 6.3.2 and output the synthesized view as the nearest neighbor (NN) image. At test time, the top match from image retrieval is used as the NN image.

Within the *Pose Regression Module*, we first process the images by a global descriptor network, *i.e.*, the feature maps from a CNN such as ResNet [122] plus global average pooling to obtain embeddings $(\mathbf{f}_{query},\ \mathbf{f}_{NN})^\top \in \mathbb{R}^{2,c}$, where c is the feature size. Inspired by recent success [77], we use a Transformer encoder [334] as our relative pose regression module. We validate experimentally the superiority of the Transformer against a MLP architecture in Section 6.4.2.

A learnt positional encoding with the input $(0,1)^\top \rightarrow (\mathbf{p}_{query}, \mathbf{p}_{NN})^\top \in \mathbb{R}^{2,d}$ is concatenated with the embeddings to form the input vectors $\in \mathbb{R}^{2,c+d}$ to the *Relative Pose Regression Transformer* (more architectural details in Section 6.3.5 in *Relative Regression Module*). The outputs of the Transformer are the relative rotation in unit quaternion $q_{rel} \in \mathbb{R}^4$ and relative translation $t_{rel} \in \mathbb{R}^3$.

To localize the query, we first convert quaternion q_{rel} to rotation matrix $R_{rel} \in \mathbb{SO}(3)$. The absolute pose prediction is given by $R = R_{NN}R_{rel}^\top$ and $t = t_{NN} + t_{rel}$, where $(R_{NN},\ t_{NN})$ is the pose of the nearest neighbor.

6.3.4 Loss

We use a weighted l_1-loss to supervize the relative pose predictions

$$\mathcal{L} = \|q_{rel} - q_{rel}^*\|_1 e^\beta + \|t_{rel} - t_{rel}^*\|_1 e^\gamma, \tag{6.3}$$

where q_{rel}^* is the relative unit quaternion between the query image and the nearest neighbor converted from the rotation matrix $R_{rel}^* = R^{*\top} R_{NN}$ and the relative translation $t_{rel}^* = t^* - t_{NN}$. $(R^*,\ t^*)$

denotes the ground-truth pose of the query. β, γ are constants for weighting between the rotation and translation terms.

6.3.5 Implementation Details

Pairs Selection - Real Data. We use directly the GeM [259] rankings to form our pairs. Given a training image, we resize it to 180 pixels on the shorter side while maintaining the aspect ratio, and extract the GeM descriptor. The top N NN are retrieved by computing the dot product of the query to the remaining training images' GeM descriptors. The query and a randomly sampled N.N from the top N neighbors form our training pair. For outdoor data, we set $N = 20$, meaning a training pair is randomly formed from a query and any of its 20 NN. In indoor data, due to the smaller scale of the scenes we train on, we allow all of the images to form training pairs with each other. In *Street*, where GeM descriptors retrieve in some cases incorrect NN, we remove those training pairs which camera position that have a L1-distance above 20 meters from the query.

In-Distribution. The selection procedure for training pairs follows the method described in *Pairs Selection - Real Data*. After obtaining a query-NN pair, we perturb the camera pose of the NN with $\alpha_q = 0.02$, $\alpha_t = 1$ m for outdoor data and $\alpha_q = 0.02$, $\alpha_t = 0.1$ m for indoor data. The query is also augmented with a rotation-only perturbation of $\alpha_q = 0.02$. For both the NN and the query, we use a probability of 50% of perturbing the poses and 50% of using the original real poses and images.

Out-of-Distribution. Given a query image, we produce a synthetic pose by adding Gaussian perturbations to the query pose C_q. For the rotation perturbation, we use the standard yaw, roll and pitch axes definitions. For each of these axes, we choose an angle ϕ by sampling from a Gaussian distribution, *e.g.*, for the yaw axis we select an angle ϕ_y by sampling from $\mathcal{N}(\mathbf{0}, \sigma_y)$. We then transform these sampled angles ϕ_y, ϕ_r, ϕ_p to a rotation matrix and use it to transform the original rotation in C_q to a new synthetic rotation. For the translations, we follow a two-stage approach. We first perturb the original translation in C_q with a larger perturbation sampled from a Gaussian with standard deviation $\alpha_{t,large}$ to form our perturbed translation. To avoid a situation in which after this large scale perturbation our pose is behind a wall, or any other *out-of-bounds* situation, we select the closest training pose

to the perturbed translation. To this closest training pose, we apply a smaller translation perturbation with standard deviation $\alpha_{t,small}$ to form our final synthetic translation. Even though this step does not fully avoid the *out-of-bounds* situation, it does reduce the likelihood of it happening. To deal with the *out-of-bounds* cases that may still occur, we remove from the training loss the contribution of those synthetic images with less than 30% available pixels, as mentioned in the implementation details given in Section 6.3.5. We select the standard deviations of the perturbations we use by inspecting the training data. For outdoor data, we notice that most of the rotations are around the yaw axis, hence we set our the standard deviations of our rotation perturbations to $\sigma_y = 30°$ around the yaw axis, $\sigma_r = 10°$ around the roll axis and $\sigma_p = 2.5°$ around the pitch axis. For the translation, we set $\alpha_{t,large} = 10$ m in the ground-plane, 0.1 m in the height axis, and $\alpha_{t,small}$ to 0.5 m in the ground plane and 0.1 m in the height axis. For indoor data, we assume the distribution of rotations is independent of the axis, hence we use a standard deviation of $15°$ in all directions, *i.e.*, $\sigma_y = \sigma_r = \sigma_p = 15°$. For the indoor translation perturbations, we set $\alpha_{t,large} = 0.5$ m and $\alpha_{t,large} = 0.25$ m in all directions. After we have created our synthetic NN pose by perturbing the query pose, we synthesize the new view by selecting the closest 10 images in translation (among those that are within a rotation threshold of $15°/30°$ for indoor/outdoor data) in the training set as the source images. We generate synthetic views with these, iteratively project from the depth maps of the NNs until either 10 images are sourced or 80% of the synthetic image pixels are filled. For each element in the batch, we randomly select with a probability of 25% the real NN chosen following the method explained in *Pairs Selection - Real Data*, instead of generating synthetically the NN. We also augment the real query image using the same rotation perturbation used in the *In-Distribution* approach, with $\alpha_q = 0.02$. As some of the generated images contain few projected pixels, we do not compute any loss for those images with less than 30% of pixels projected. During the first 700 epochs of training, we randomly assign the synthetic view as either the NN or query with a probability of 50%, whereas we fix the synthetic view as the NN after 700 epochs, which we found experimentally to increase the performance.

Relative Regression Module. Our CNN for the relative regression model is a ResNet-34 [122] pretrained in ImageNet [71]. The feature embedding of dimensionality 512 output by the ResNet-34 model is passed to our *Relative Pose Regression Transformer* [334], which is shown in Figure 6.5. Our Transformer encoder receives a sequence of two embeddings of dimensionality 512,

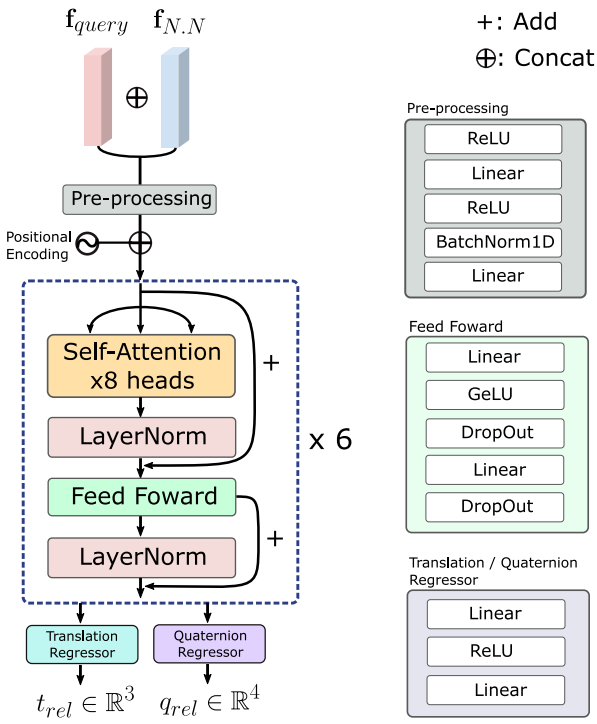


Figure 6.5: Architecture of our *Pose Regression Transformer*.

the query $\mathbf{f}_{query}$ and the NN embedding $\mathbf{f}_{NN}$, which are first passed to a *Pre-processing* module (`[Linear(512,2048), ReLU, BatchNorm1D, Linear(2048, 512)]`) before being fed to a Transformer encoder that contains 6 layers and 8 heads, where each head has a dimensionality of 128. The dropout used in the *Feed Forward* modules is set to 0.1. Before the two vectors are fed to the Transformer encoder, we concatenate to both of them a positional encoding, either $\mathbf{p}_{query}$ or $\mathbf{p}_{N.N}$, of size 128 that allows the Transformer to know which embedding comes from the query and which from the NN. Our Transformer encoder thus receives embeddings of size $512 + 128 = 640$. The Transformer output corresponding to the position of the NN is then fed to two different regressors, the *Translation Regressor* (`[Linear(640,640), ReLU, Linear(640, 3)]`) and the *Quaternion Regressor* (`[Linear(640,640), ReLU, Linear(640, 4)]`) to output t_{rel} and q_{rel} respectively. Due to issues with batch normalization, we switch the networks to evaluation mode after the first learning rate drop at 500 epochs, thus after the first 500 epochs the batch normalization layers use the batch statistics (*i.e.*, mean and variance) learned during the first 500 epochs during training. We found experimentally that this approach offers a slight performance increase. We hypothesize this effect is related to merging two different distributions in the batch during training, *i.e.*real and synthetic images [341]. Furthermore, the variable density of available pixels in the synthetic images may

Table 6.2: Comparison of visual localization methods on *Cambridge Landmarks* [154]. We report the median errors in translation (m) / rotation (°). The four types of methods are image retrieval (IR), structure-based (3D), absolute pose regression (APR) and relative pose regression (RPR), which our method belongs to. We highlight the best results for IR, 3D and all regression methods with red, blue and bold respectively. [(-) no results reported (✗) method fails to train.]

		Cambridge Landmarks				
		King's College	Old Hospital	Shop Façade	St. Mary's Church	Street
IR	DenseVLAD [326]	2.80/5.72	4.01/7.13	1.11/7.61	2.31/8.00	5.16/23.5
IR	DenseVLAD + Inter. [296]	1.48/4.45	2.68/4.63	0.90/4.32	1.62/6.06	15.4/25.7
IR	GeM [260]	3.22/5.71	4.46/7.11	1.57/8.25	3.01/9.37	6.39/23.6
3D	Active Search [292]	0.42/0.55	0.44/1.01	0.12/0.40	0.19/0.54	0.85/0.8
3D	DSAC++ [28]	0.18/0.3	0.20/0.3	0.06/0.3	0.13/0.4	✗
APR	PoseNet [154]	1.92/5.40	2.31/5.38	1.46/8.08	2.65/8.48	-
APR	Geometric PoseNet [153]	0.88/1.04	3.20/3.29	0.88/3.78	1.57/3.32	20.3/25.5
APR	PoseNet LSTM [338]	0.99/3.65	1.51/4.29	1.18/7.44	1.52/6.68	✗
APR	MapNet [30]	1.07/1.89	1.94/3.91	1.49/4.22	2.00/4.53	-
APR	GRNet [354]	0.59/0.65	1.88/2.78	0.50/2.87	1.90/3.29	14.7/22.4
RPR	AnchorNet [283]	0.57/0.88	1.21/2.55	0.52/2.27	1.04/2.69	7.9/24.2
RPR	Ours - *In-Distribution*	**0.24/0.38**	0.83/1.05	**0.26**/1.06	0.58/1.56	5.81/16.2
RPR	Ours - *Out-of-Distribution*	0.35/0.51	**0.65/0.93**	0.32/**0.94**	**0.43/1.18**	**3.85/7.18**

result in the batch statistics having a high variance between different batches due to the relatively small batch size used (16 image pairs per batch), which can affect the training process [309].

Training. For each scene, we train the *Pose Regression Module* with the loss in Equation (6.3) for 1200 epochs with batch size of 16. We optimie using Adam [162] at an initial learning rate of 10^{-4}, with a decay rate of 0.1 every 500 epochs. Our code is implemented using PyTorch and we use Kornia for our view-synthesis pipeline [268]. Images are resized to 180 pixels on the shorter side (resulting in images of size 320×180 in *Cambridge Landmarks* and 240×180 in *7Scenes*) and random color-jitter is applied. For the loss in Equation (6.3), (β, γ) are set to $(3, 0)$ for outdoor and $(0, 0)$ for indoor scenes.

Inference. For each test query, we retrieve the nearest neighbor from the training set with GeM. Images are resized to 180 pixels on the shorter side (*i.e.*, images of 320×180 in *Cambridge Landmarks* and 240×180 in *7Scenes*) but without color-jitter. The inference speed is over 300 frames per second on a Titan RTX, which is high, mainly due to the small size of the images we use. Detailed timing breakdown and comparisons with other methods are given in Section 6.4.3.

Table 6.3: Comparison of visual localization methods on *7Scenes* [103]. We report the median errors in translation (m) / rotation (°). The four types of methods are Image Retrieval (IR), structure-based (3D), Absolute Pose Regression (APR) and Relative Pose Regression (RPR), which our method belongs to. We highlight the best results for IR, 3D and all regression methods with red, blue and bold respectively.

		7Scenes						
		Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
IR	DenseVLAD [326]	0.21/12.5	0.33/13.8	0.15/14.9	0.28/11.2	0.31/11.3	0.30/12.3	0.25/15.8
IR	DenseVLAD + Inter. [296]	0.18/10.0	0.33/12.4	0.14/14.3	0.25/10.1	0.26/9.42	0.27/11.1	0.24/14.7
IR	GeM [260]	0.27/13.1	0.39/15.8	0.21/15.7	0.37/11.7	0.43/11.6	0.42/13.4	0.34/14.2
3D	Active Search [292]	0.04/1.96	0.03/1.53	0.02/1.45	0.09/3.61	0.08/3.10	0.07/3.37	0.03/2.22
3D	DSAC++ [28]	0.02/0.5	0.02/0.9	0.01/0.8	0.03/0.7	0.04/1.1	0.04/1.1	0.09/2.6
APR	PoseNet [154]	0.32/8.12	0.47/14.4	0.29/12.0	0.48/7.68	0.47/8.42	0.59/8.64	0.47/13.8
APR	Geometric PoseNet [153]	0.13/4.48	0.27/11.3	0.17/13.0	0.19/5.55	0.26/4.75	0.23/5.35	0.35/12.4
APR	PoseNet LSTM [338]	0.24/5.77	0.34/11.9	0.21/13.7	0.30/8.08	0.33/7.00	0.37/8.83	0.40/13.7
APR	MapNet [30]	0.08/3.25	0.27/11.7	0.18/13.3	0.17/5.15	0.22/4.02	0.23/4.93	0.30/12.1
APR	GRNet [354]	0.08/2.82	0.26/8.94	0.17/11.4	0.18/5.08	0.15/2.77	0.25/4.48	0.23/8.87
RPR	RelocNet [17] (SN)	0.21/10.9	0.32/11.8	0.15/13.4	0.31/10.3	0.40/10.9	0.33/10.3	0.33/11.4
RPR	RelocNet [17] (7S)	0.12/4.14	0.26/10.4	0.14/10.5	0.18/5.32	0.26/4.17	0.23/5.08	0.28/7.53
RPR	AnchorNet [283]	0.06/3.89	0.15/10.3	0.08/10.9	0.09/5.15	0.10/2.97	0.08/4.68	0.10/9.26
RPR	CamNet [75]	0.04/1.73	**0.03**/1.74	0.05/**1.98**	**0.04**/1.62	**0.04**/1.64	**0.04**/1.63	**0.04**/1.51
RPR	Ours - *In-Distribution*	0.04/1.69	0.13/5.22	0.08/5.95	0.09/2.70	0.12/2.48	0.11/2.93	0.26/5.13
RPR	Ours - *Out-of-Distribution*	**0.03/0.92**	0.04/**1.32**	**0.04**/2.78	**0.04/1.35**	**0.04/0.98**	**0.04/1.45**	0.07/**1.44**

6.4 Results

We now present and discuss the visual localization results on the two main benchmarks for pose regression: *Cambridge Landmarks* [154] and *7Scenes* [103] (see Section 2.6.3 for an explanation of the two datasets). Regarding the depth maps used for view sythesis, for the *Cambridge Landmarks* dataset, we generate MVS depth maps in metric scale using COLMAP [302] and the Kapture [138] pipeline with AP-GeM [266] and R2D2 [267] (see Section 2.5 for MVS explanation). In *7Scenes*, the Kinect depth given in the dataset is not used as it is not aligned with the RGB images, hence we use the depth maps given in the official code of DSAC* [29], which were computed using a 3D model.

Evaluation Metrics. The rotation error and the translation error are the main evaluation metrics. For the translation error, given two 3DoF poses (t_x, t_y, t_z), we can compute the l_2 norm between the ground truth and the predicted pose, *i.e.*, the translation error is $\sqrt{(t_x - t_x^*)^2 + (t_y - t_x^*)^2 + (t_z - t_z^*)^2}$. As for the rotation error, we use $2 \arccos(|\langle q, q^* \rangle|)$ to compute the error between the predicted and ground-truth quaternions, where $\langle q, q^* \rangle$ is the inner product defined as $q_1 q_1^* + q_2 q_2^* + q_3 q_3^* + q_4 q_4^*$. We

report the median translation and rotation errors for the test set.

6.4.1 Comparison with State-of-the-Art

We compare the performance of our pipeline with existing methods in Table 6.2 and Table 6.3. As Sattler *et al.* [296] pointed out, both datasets are challenging for pose regression methods as the training and test poses are sampled from separate sequences and the trajectories differ significantly. Following [296], we include all four major types of localization methods: Image Retrieval (IR), structure-based (3D), Absolute Pose Regression (APR) and Relative Pose Regression (RPR).

From Table 6.2, our method gives the lowest median errors in the Cambridge dataset, by a large margin, when compared to all other existing pose regression methods, including both APR and RPR. On the *King's College* scene in *Cambridge Landmarks* [154], our *In-Distribution* variant gives (0.24m/0.38°) which outperforms even *Active Search* [292] (0.42m/0.55°), that was a long-standing point of reference when comparing APR and RPR methods' validity against 3D and IR. It also comes very close to *DSAC++* [28] (0.18m/0.3°). To our best knowledge, this the first pose regression method that outperforms Active Search on a *Cambridge Landmarks* scene.

On *Street*, a scene where no previous regression methods have improved upon image retrieval, our *In-Distribution* and *Out-of-Distribution* improve upon the retrieval with GeM [260], which we use in our pipeline as well as upon the state-of-the-art retrieval with DenseVLAD [296, 326].

The results for *Out-of-Distribution* are significantly better than for all other regression methods, reducing the median errors in translation and rotation by 45% and 60%, respectively when compared to the state-of-the-art in RPR AnchorNet [283]. The median errors of *Out-of-Distribution* on *Old* and *St. Mary's* are even lower than that of *In-Distribution*. These two scenes are the more challenging ones amongst the smaller scenes in *Cambridge Landmarks* [154], as suggested by their relatively higher median errors in all existing regression methods when compared to *King's College* and *Shop Façade*.

These results show that our *Out-of-Distribution* approach generalizes well to all scales and compositions of scenes, and it is robust to data biases, such as in *Street*. Conversely, *In-Distribution* is suitable when the training and test distribution are more aligned, *e.g.*, *King's College* and *Shop*.

Table 6.4: Ablation study with respect to different components of our localization pipeline. We compare training with real image pairs only and our *In-Distribution* approach. For each case, we also compare the use of MLP vs. Transformer.

Training	Arch.	King's College	Old Hospital	Shop Façade	St. Mary's Church	Street
Real	MLP	0.46/0.72	2.19/2.75	0.55/2.06	1.48/3.35	-
	Trans.	0.45/0.63	1.62/2.60	0.54/2.05	1.28/2.99	-
Synthetic - *In Dist.*	MLP	0.26/0.45	**0.81**/1.37	0.29/1.16	0.73/1.95	6.0/18.6
	Trans.	**0.24/0.38**	0.83/**1.05**	**0.26/1.06**	**0.58/1.56**	**5.8/16.2**

On *7Scenes*, as Table 6.3 shows, both our *In-Distribution* and especially *Out-of-Distribution* method again outperform most regression methods significantly. Comparing to the best indoor RPR method CamNet [75], our *Out-of-Distribution* results match in translation errors and mostly outperform in rotations. However, it is to be noted that CamNet, additionally from using scene-specific retrieval, uses multiple branches for coarse-to-fine pose predictions and predictions from multiple nearest neighbors, while our methods only require a single query-NN prediction at inference.

We compare the average median errors on *Cambridge Landmarks* scenes (with the exception of *Street*) of *In-Distribution* (0.48m/1.01°) and *Out-of-Distribution* (0.44m/0.89°) to *Active Search* [292] (0.29m/0.63°) and *DenseVLAD + Interpolation* [296] (1.67m/4.87°). The gap between the errors of our methods and that of *Active Search* (≈0.2m/0.4°) is 6-10 times less than that to *DenseVLAD + Interpolation* (≈1.2m/4°). Similar observations are made from the results for *7Scenes*. This validates our approach and supports our claim that our regression method is the first to achieve a performance closer to structure-based methods than to image retrieval for both outdoor and indoor data.

6.4.2 Ablation Studies

In this section, we analyze the respective impact of the three main components of our approach.

Training with Real vs. Synthetic Views. From Table 6.4, we observe that the median errors in *Cambridge Landmarks* are significantly reduced by about 50% when training with synthetic views by *In-Distribution* sampling compared to using only real images. In order to verify whether the gain in performance is solely due to the increased number of training samples, or if the use of synthesized

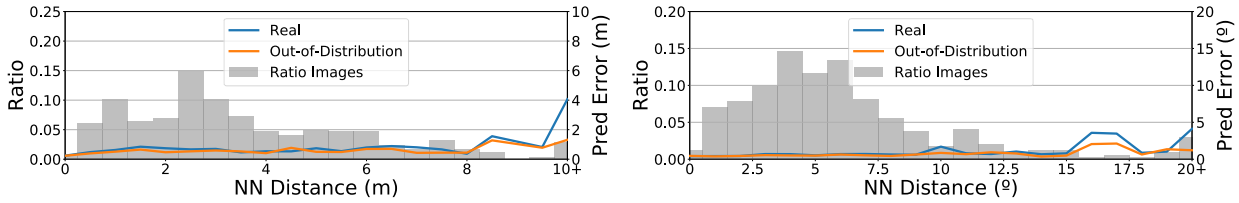
Table 6.5: Results when using limited amount of data jointly with our synthetic images. We use *Out-of-Distribution* to generate a single novel view per training image for different amounts of data, and we use the resulting pairs for training.

		% of training images				
Scene	# images	All real	10	25	50	Out-of-Dist.
Chess	4000	0.13/5.25	0.11/4.32	0.08/2.88	0.06/2.15	0.03/1.10
Heads	1000	0.17/13.7	0.15/12.4	0.14/11.3	0.13/9.81	0.04/2.78

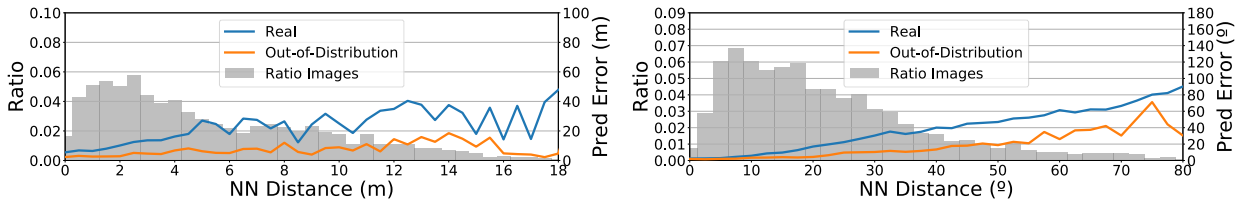
views truly aligns the training and test distributions, we conduct an experiment which is presented in Table 6.5. We choose two scenes from *7Scenes* with different number of training images. For each scene, we sample a ratio of training images, which we use to generate a single synthetic view each, using our *Out-of-Distribution* method to form our training pairs. For example, in *Chess*, which contains 4000 images, the 50% result means that 2000 real training images are selected, each paired with a synthetic view, adding up to a total of 4000 training images. For 25% that would be 1000 real and synthetic images each for a total of 2000, and so on. Table 6.5 shows for both scenes that even when training with only 10% of the original real images and their corresponding generated synthetic views, we obtain a higher performance than when using all the available real images. This result shows that the distribution of the relative poses is more important than the number of images used.

MLP vs. Transformer. We also compare the MLP used in prior RPR methods [75, 283], which predicts the relative pose from two concatenated image embeddings, to the Transformer we use in our *Pose Regression Module*. From Table 6.4 we see that although the overall performance boost of our localization pipeline can be largely attributed to training with synthetic views, using the Transformer still gives a notable improvement of the median errors.

Performance vs. NN Distance. Figure 6.6 includes the error of the predicted pose against the distance of the NN with respect to the query for two different scenes. We separate in Figure 6.6 translation and rotation errors in two different plots, and we include the error for a model trained only with real images and a model trained with *Out-of-Distribution* method. The images also include a histogram of the ratio of query-NN pairs that are at a specific distance. As expected, in all the presented



(a) *King's College*



(b) *Street*

Figure 6.6: Median error in predictions depending on the distance to the NN for two different scenes (*King's College* and *Street*) and for the models trained only with real data and the models trained with our *Out-of-Distribution* method. We include both rotation and translation errors, along with the ratio of test query-NN pairs (shown in *Ratio Images*) for the different distances shown.

plots, closer NNs to the query lead to a lower error for the predicted query pose. For *King's College* we observe how the predicted error between the *Real* and *Out-of-Distribution* model is close for lower NN distances. However, due to the narrow distribution of poses in the database images (*e.g.*, Figure 6.2 shows the narrow spread in the yaw angle for this scene) if we only sample our training pairs from real data the model will struggle with test query-NN pairs showing a larger viewpoint change, which is shown in Figure 6.6 as for the larger errors *Out-of-Distribution* consistently outperforms the model trained only with real data in *King's College*. For *Street*, our *Out-of-Distribution* method achieves a consistent improvement for all ranges compared to the model trained only with real data. We hypothesize that this consistent improvement is related to the bias in the poses shown in Figure 6.2, as sampling training pairs from that distribution will result in the model only learning from a limited range of relative poses.

Closest Training Synthetic Pose. Table 6.6 shows the median distance of the test queries to the closest synthetic pose generated during training. For the smaller scenes (*i.e.*, all but *Street* in Table 6.6), the median distance to the closest synthetic pose seen during training is higher than the reported errors for both our *In-Distribution* and *Out-of-Distribution* methods in Table 6.2. These results show that our method is capable of interpolating over the seen synthetic poses to further reduce the error, and

Table 6.6: Median distance between the test query images and the closest synthetic pose generated during training. We generate a synthetic pose following either our *In-Distribution* or our *Out-of-Distribution* approach for 1200 epochs following our experimental settings. To retrieve the closest synthetic pose, we use as metric a weighted sum of translation and rotation distance (*i.e.*, $\lambda t_d + r_d$), where the weight λ is obtained using the ratio of rotation and translation errors in Table 6.2 for each scene and method. For example, for the reported closest synthetic pose for *Out-of-Distribution King's College* the weight λ is set to $\lambda = \frac{0.51}{0.35}$.

	King's College	Old Hospital	Shop Façade	St. Mary's Church	Street
In-Distribution	0.76/1.03	1.51/1.65	0.52/1.93	0.74/2.07	1.56/11.00
Out-of-Distribution	0.91/1.34	1.31/1.31	0.62/1.68	0.65/2.00	0.92/2.07

Table 6.7: Additional quantitative results in *7Scenes* [103]. *Real* refers to training with only real data, and we include results for our *Out-of-Distribution* method (*O.o.D*) when using during inference either the GeM top-1 NN or a randomly chosen database image as our NN. Reported results are the median translation (m) / rotation (°) errors.

	7Scenes						
	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
Real	0.13/5.25	0.32/13.4	0.17/13.7	0.24/7.10	0.26/6.25	0.27/6.77	0.30/13.3
O.o.D.+Random	0.03/1.20	0.04/2.07	0.05/3.37	0.04/1.41	0.04/1.24	0.06/1.71	0.09/1.79
O.o.D.+GeM	0.03/1.10	0.04/1.64	0.04/2.78	0.04/1.35	0.04/1.20	0.05/1.62	0.10/1.62

is thus not merely memorizing the poses seen during training. For *Street*, which is a larger scene, the median distance to the closest training synthetic pose is lower than the errors reported in Table 6.2 for our methods. We hypothesize that a better method to sample training query-NN pairs and an increased quality of synthetic images could further improve the performance on *Street*.

Additional Results in 7Scenes are included in Table 6.7. Due to the scenes available in *7Scenes* being constrained to a small volume, we allowed all the available real images for each scene to form training pairs with each other. *Real* shows the performance when training only with real pairs, which shows that our network overfits quickly to the training set, and generalizes poorly to the test set as the training poses come from a limited set of video sequences. Our *Out-of-Distribution* method, which forms training pairs by using a Gaussian perturbation based approach that allows the network to learn from less biased relative poses, improves upon the *Real* results by a large margin in all cases as we see in *O.o.D.+GeM* in Table 6.7. As the *Out-of-Distribution* approach trains from a wide range of

Table 6.8: Results in *Shop Façade* and *Shop Façade Synthetic*, along with the *Relative Performance* difference between evaluating on *Shop Façade* vs. *Shop Façade Synthetic*. *Baseline Synth.* is our method trained with only with synthetic images from our *Out-of-Distribution* module without multi-image accumulation (*i.e.*, only projecting a single database image) and only using COLMAP depth. + *Multi-image accumulation* adds to *Baseline Synth.* the accumulation of 10 database images to generate the synthetic images. + *Fused Depth* further adds to *Baseline Synth.* + *Multi-image accumulation* a fusion of the predicted depth with the COLMAP depth to generate denser depth maps. + *Real (Full)* is our full *Out-of-Distribution* pipeline, which combines 25% of fully real images to the training batch with the synthetic images generated with multi-image accumulation and depth fusion.

	Baseline Synth.	+ Multi-image Accumulation	+ Fused Depth	+ Real (Full)
Shop Façade Synthetic	0.57/2.19	0.36/1.60	0.28/1.00	0.28/1.05
Shop Façade	0.70/2.42	0.44/1.34	0.34/1.33	0.32/0.94
Relative Difference	↑ 22.8%/ ↑10.5%	↑22.2%/↓18.3%	↑21.4%/↑33.0%	↑14.3%/↓10.5%

distances and rotations, we explore in the line *O.o.D.+Random* the impact of randomly chosing a nearest neighbor during inference instead of using the top-1 GeM NN. Table 6.7 shows that a similar performance regardless of the NN used, which shows the robustness of our approach to the NN choice in smaller scenes.

Closing the Domain Gap. In Table 6.8 we test the effect of the different modules in our pipeline that aim to improve the performance on real data when training with synthetic images. To explore the domain gap we create *Shop Façade Synthetic* by generating for each of the poses in the test set of *Shop Façade* a synthetic image using the projection module presented in Section 6.3.2 (with fused depth and multi-image accumulation). Hence, *Shop Façade Synthetic* follows a similar image distribution to the synthetic training images when using our *Out-of-Distribution* module. Our baseline method fully trained with synthetic images, *i.e.*, *Baseline Synth* in Table 6.8, presents better performance for both rotation and translation in *Shop Façade Synthetic* compared to *Shop Façade*, where the error is considerably larger than the error obtained with our full pipeline (+ *Real (Full)*). Increasing the synthetic image quality (both + *Multi-image Accumulation* and + *Fused Depth*) improves the performance in both datasets. However, the rotation error seems to decrease more for the real *Shop Façade* when adding + *Multi-image Accumulation* and more in *Shop Façade Synthetic* when adding + *Fused Depth*, as observed in the relative difference in performance between the two datasets. The lower improvement on rotation error of + *Fused Depth* for the real *Shop Façade* could be due to the errors

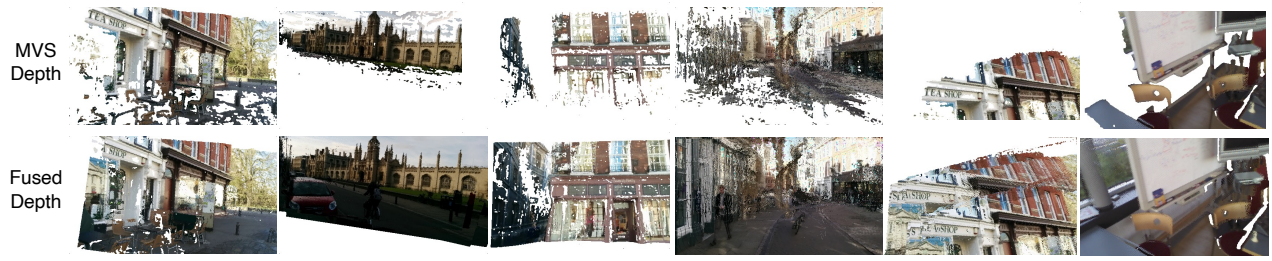
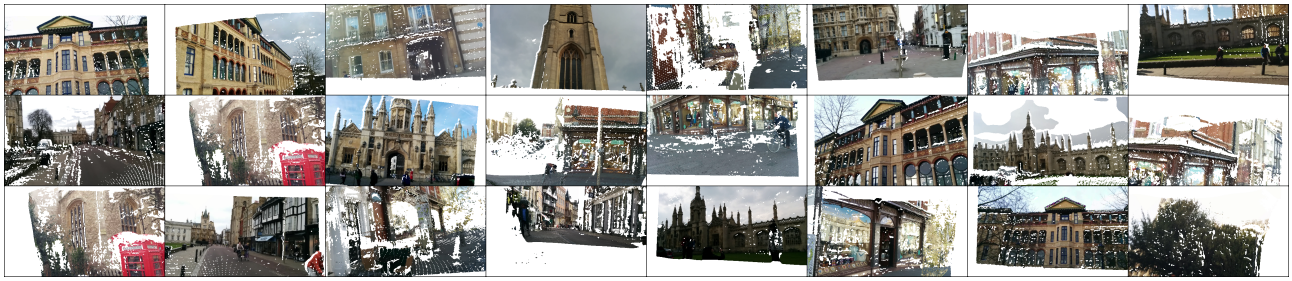


Figure 6.7: Examples of novel views synthesized with our approach. Using only the *MVS Depth* limits the quality of the synthesized image due to the depth sparsity. Our *Fused Depth*, presented in Section 6.3.2, allows for denser synthetic views.



(a) Synthetic images generated by using our *Out-of-Distribution* approach in outdoor scenes

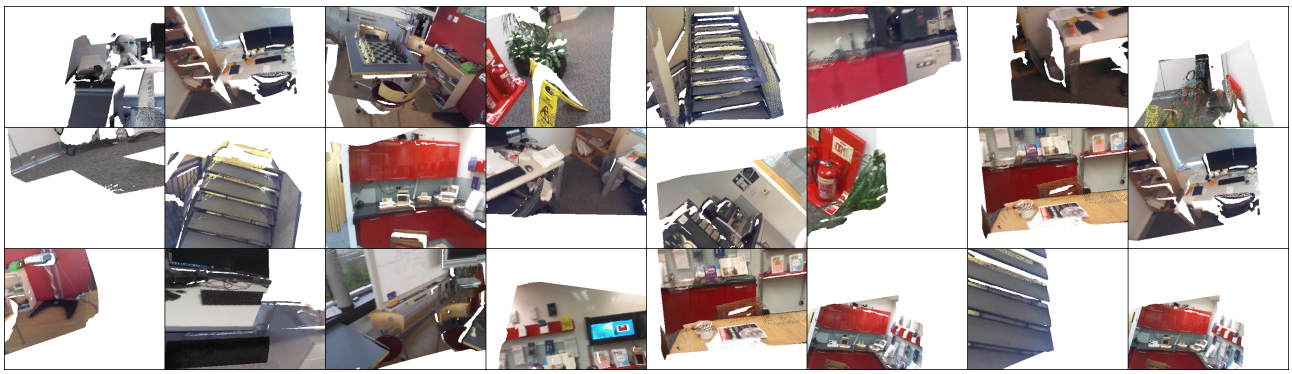


(b) Synthetic images by using our *In-Distribution* approach in outdoor scenes

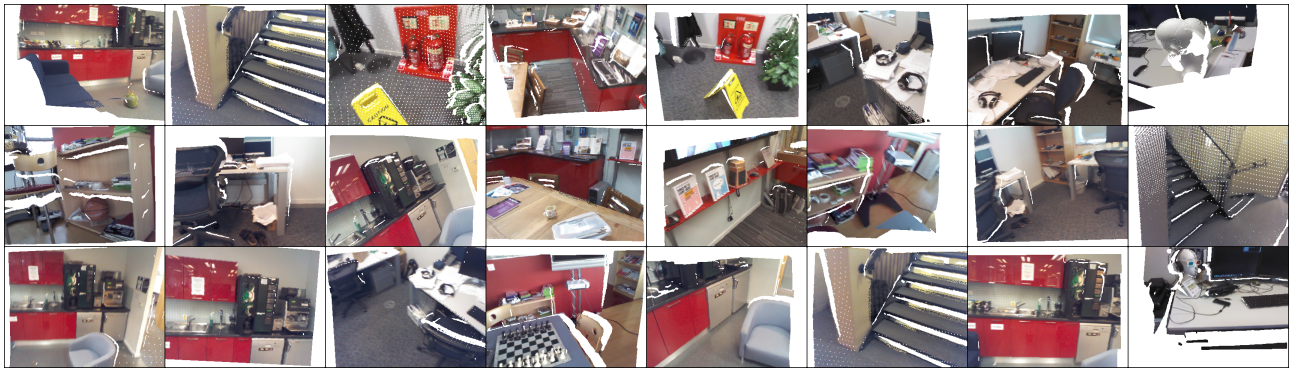
Figure 6.8: Synthetic images used during training in the Cambridge dataset. The images are randomly selected from our method, we filtered the images with less than 30% non-blank pixels.

in the predicted depth adding noise when using the fused depth. Finally, introducing real query-NN training pairs (+ *Real (Full)*) has the aim of reducing the domain gap by allowing the network to learn from both real and synthetic images at the same time. Hence, + *Real (Full)* improves the performance in *Shop Façade* whereas it does not improve the results in *Shop Façade Synthetic*. Our full pipeline reaches a similar error in both *Shop Façade* and *Shop Façade Synthetic*.

Quality of Synthetic Images. In Figure 6.7 we show examples of the synthetic images we use for training, where we compare using only MVS depth against our depth fusion module. Synthetic images generated from MVS depths are sparser, showing many holes and flying pixels. However, the fusion of predicted dense and sparse MVS depths is able to produce much denser synthetic views.



(a) Synthetic images generated by using our *Out-of-Distribution* approach in indoor scenes



(b) Synthetic images generated by using our *In-Distribution* approach in indoor scenes

Figure 6.9: Synthetic images used during training in the 7Scenes dataset. The images are randomly selected from our method, we filtered the images with less than 30% non-blank pixels.

Compared to the MVS depths, which have been generated under a multi-view consistency manner, the predicted depths have lower accuracy, which is manifested by the flying pixels or stitching issues present in some views, as shown in *e.g.*, the fourth and fifth columns in Figure 6.7. The depth maps from the indoors scenes we use are denser, hence the effect of fusing the depths is less notable. The quality of synthetic images vary, but even so, our network is capable of learning from such views and generalizing to the real images in the test data. This is a significant observation, since it shows that due to the holistic nature of the pose regression methods, images with missing or noisy local regions, are still able to contribute to localization. Figure 6.8 and Figure 6.9 show some further examples of synthetic views for both indoor and outdoor data for both our *In-Distribution* and our *Out-of-Distribution* approach. We removed the images with less than 30% of projected pixels as they are not taken into account to compute the loss. The indoor synthetic views, due to the larger density of views and the more constrained scenes, present usually a higher quality of synthesized views.

Qualitative Results. Figure 6.10 shows some qualitative results of the predictions obtained using



Figure 6.10: Qualitative results with our *Out-of-Distribution* trained model. For visualization purposes, we synthesize an image in the pose predicted by our model (*Prediction* columns).

our *Out-of-Distribution* trained model. We also include the NN retrieved by GeM used to compute the relative pose. In the smaller scenes (*i.e.*, top two examples in Figure 6.10, corresponding to *Shop Façade* and *King's College*), the model can predict highly accurate poses, shown by the high similarity between the query image and the synthesized image in the predicted pose. For *Street* (bottom two images), the predictions are less accurate than in the smaller scenes, even for the cases when NN and the query images are close (bottom-left example in Figure 6.10). The retrieved NN usually contains a high covisibility with the query image, however our model is also capable of showing an understanding of the scene for NNs with lower covisibility (bottom right example in Figure 6.10). .

6.4.3 Inference Speed

We use for our experiments a NVIDIA Titan RTX GPU, 64GB of RAM, a NVMe SSD, and a i9-10980XE CPU at 3.00 GHz. We assume that the embeddings used for retrieval for the training images are already precomputed, and during inference we only need to compute the GeM embedding for the query images. Similarly, we also assume our $\mathbf{f}_{N.N}$, which are shown in Figure 6.5, are also precomputed. For these tests we use a batch size of 16, and unless stated otherwise we ran the tests in the *Cambridge Landmarks* [154] dataset with input image size of 180×320.

Retrieval. We use a pretrained GeM ResNet-50 network for retrieval. The inference time for the retrieval network for a batch of 16 images of size 180×320 is 21 ms, or 762 frames per second.

Relative Pose estimation can be done in 20 ms, or 800 frames per second, for a batch of 16 images of size 180×320. We assume the features for the database images are already precomputed.



Figure 6.11: Examples of incorrect NNs retrieved by GeM in *Street*.



Figure 6.12: Examples of incorrect predictions in *Street*. For visualization purposes, we synthesize an image in the pose predicted by our model (*Prediction* columns).

Total. Our final method needs two forward passes, one for the retrieval network, and one for the relative pose estimation network. Hence, our total computational time per batch is of 41 ms, which translates to 390 frames per second. However, to the given estimates we need to add the effects of data loading, NN search and the transformation from relative to absolute pose. The whole pipeline computes the poses for the 343 images in the test set of *King's College* (*Cambridge Landmarks* dataset) in 1.10 s using a dataloader with 16 workers, amounting to 310 frames per second. For *7Scenes* [103], where we use an image resolution of 180×240, we compute the 1000 test poses in *7Scenes-heads* in 2.75 s, amounting to 364 frames per second. Additionally, in *7Scenes*, we obtain similar results by choosing a random database image as NN (*i.e.*, skipping the retrieval step) as shown in Table 6.7, which makes our method capable of computing the poses in *heads* in 1.90 s, or 526 frames per second.

Comparison. Our inference time shows that our method is fast compared to 3D-based methods. Even though the hardware is different, in DSAC++ [28] computing a query pose takes 200ms per image, or 5 frames per second. ActiveSearch [292] is capable of computing the pose for a query image in 36ms, or 28 frames per second. In both cases our method is considerably faster. Compared to the *7Scenes* state-of-the-art, CamNet [75], their base approach uses two relative pose predictions (coarse+fine) from two different NNs, and their final results include a RANSAC approach applied to the pose predictions that requires predicting the relative pose from 5 different NNs.

6.4.4 Limitations

Firstly, our method is limited by the quality of the synthetic data. Even though we have shown successful learning capacity from the generated synthetic images, improved synthetic views would further close the gap with structure-based methods. Our method is also limited by the quality of the retrieved NN, as we use an off-the-shelf retrieval method along with the top-1 NN at testing time without any further verification of correctness. Figure 6.11 shows some examples of incorrect NN, which will also lead to an inaccurate pose prediction. We also include some inaccurate predictions on *Street* in Figure 6.12. Related to the incorrect NN issue, the top-left example in Figure 6.12 shows that the network tries to match the rotation and closeness to the wall of the query with respect to the incorrect NN. Including a confidence measure in the predicted pose may help to filter out those cases when the retrieved NN is not correct. Furthermore, our approach is not able to deal with more extreme pose variations with the same accuracy as structure-based methods (*e.g.*, the accuracy gap in *Street* scene, which includes large pose variations, between our method and structure-based methods), which could be improved by introducing some geometric-constraints into the regression approach. On that note, we included in Figure 6.12 an example of an inaccurate prediction for a larger pose variation in the bottom-left example. Figure 6.12 also includes in the top-right example a query-NN pair with high covisibility where classical methods are likely to produce a good prediction. Related to that, the bottom-right example in Figure 6.12 shows that due to the learning-based nature without any geometric knowledge of our method, the method incorrectly predicts in some easy cases (*i.e.*, high-covisibility and close in pose NN and query) a large displacement with respect to the NN. Relative pose regression methods also need to improve their generalization abilities as, contrary to local-features based methods [290], in the proposed approach we train a single model per scene.

6.5 Conclusion

In this work, we have exposed and addressed the problem of mismatch between the training and test distributions for regressing camera pose, which is the most important factor for the performance gap between direct pose regression and structure-based methods. We proposed a strategy to generate new

camera poses guided by the probabilistic distribution of the training poses to correct the data bias. We have introduced a novel training process that is based on new view synthesis by combining dense scale-agnostic depth with sparse metric depth. Similarly to the work presented in past chapters, we have leveraged synthetic data to improve the performance in our real data without needing to collect any extra images. We have also adapted the Transformer model and have shown its superiority to the previously used MLP for pose regression. Importantly, the presented improvements bring the performance of the regression methods closer to structure-based methods, which is a significant step towards an end-to-end approach that can eventually replace the structured ones. Our approach is the first direct regression method to achieve better results than image retrieval in a very challenging large outdoor scene.

Chapter 7

Conclusions and Future Work

Data is the main constraint in current Deep Learning pipelines, and labelling is costly and time-consuming. For that reason, we have focused on Domain Adaptation (DA) methods. In our proposed approaches, we have implemented novel ideas adapting standard DA techniques (see Section 2.1.1 for the list of techniques) to a wide range of tasks.

Data Adaptation. As the main observation from our experiments, we show that data adaptation is a promising direction in most tasks. Data adaptation usually increases the performance, for example in depth estimation, where most approaches build on it (see Section 2.4). Adversarial image adaptation is widely used and a good first choice that usually leads to good performance for RGB translation, *e.g.*, used in our depth estimation approach in Chapter 4. However, other data adaptation methods may be more beneficial for some specific settings. On that note, style transfer led to better performance for our multi-style object detection (see Chapter 3) as it solved the lack of specific style selection of the usual CycleGAN approach. Another data adaptation direction was investigated in our depth completion approach (see Chapter 5), where we injected prior task knowledge to our sparse depth generation to obtain realistic noise while preserving the underlying geometric information.

Consistency and Pseudo-labelling. Exploiting consistency is also a promising direction for a wide range of tasks. Although generic pixel-wise consistency can be beneficial for various tasks [53], we argue that the applied consistency term can be tailored to a specific task and lead to better performance. On that matter, both our multi-style local feature consistency in our object detection approach

(see Chapter 3), and our semantic-depth consistency for depth estimation (see Chapter 4) improved the performance while leveraging task-specific knowledge. Pseudo-labelling (*i.e.*, providing some sort of annotation to the target data) also proves beneficial in a wide variety of tasks if the annotations have low noise. Pseudo-labelling can be tailored to the task as well, as it was shown in our proposed use of robust pseudo-labelling for object detection (see Chapter 3), noise-filtered input LiDAR depth map for depth completion (see Chapter 5) or size-depth relationships for depth estimation (see Chapter 4).

Synthetic Data is one of the main driving forces behind the use of Domain Adaptation. Even though naive training with synthetic data may lead to low performance (see Chapter 4 and Chapter 5), DA approaches can help bridge that gap. Furthermore, we can reduce the domain gap directly if we can control the synthetic data generation. For example, we applied knowledge of the physical sensor setup to generate synthetic data similar to a real LiDAR setup, which greatly improved the performance (see Chapter 5). Another example was given in our pose estimation work, where we improved the density of training poses with synthetic images thus reducing the domain gap.

Chapter 3 - Object Detection. Focusing on the proposed research questions, the first question we aimed to answer was *How can we adapt well-known Domain Adaptation methods to a multi-style target dataset?* in an object detection context. We showed that a style-transfer approach that transfers data to individual styles is crucial for adaptation, as it guarantees representation of each individual target style in the adapted data. Explicitly constraining the higher-level CNN features to be invariant to style changes with local feature consistency, further increases the robustness of the network. Furthermore, the combination of this multi-style consistency with proposed pseudo-labelling was also proven beneficial due to the classifiers being adapted to higher-level changes.

Chapter 4 - Depth Estimation. Our depth estimation method aimed to explore multi-modality consistency to answer *Can we exploit the natural relationship between depth and semantics to guide the adaptation of a depth estimation network?* We showed that natural relationships between depth and size with semantics can be used as pseudo-labels to guide the learning process. Also, to further improve the performance, we forced consistency between depth predictions from RGB images and from semantic maps, based on the assumption that domains are more similar (*i.e.*, less domain gap) in

high-level semantic information than in the pixel-space. Our method can be combined with different state-of-the-art image transfer methods, which suggests that adaptation at multiple levels (*e.g.*, image and output-level adaptation) can be combined successfully in a depth estimation setting.

Chapter 5 - Depth Completion. Using adapted synthetic data for depth completion allowed our method to improve upon all past approaches not using real domain ground-truth. Specifically, answering our question *How important is the adaptation of the RGB and input depth modalities?*, our experiments showed that sparse depth adaptation is crucial, whereas adapting the RGB information only brings a small improvement. Using a standard adversarial RGB adaptation led to limited improvements, while the tested feature adversarial adaptation methods did not improve the results either.

Chapter 6 - Camera Pose Localization. In our pose estimation approach, we generated on-the-fly synthetic image pairs for relative pose estimation using fused depth maps and multiple image accumulation, which lead to a large increase in performance. As for the research question *Is the lower performance of CNN-regression methods compared to classical methods (partly) due to data issues (*e.g.*, bias or amount of data)?*, we found that both the bias in the training poses compared to the testing poses and the quantity of data does limit the performance. When overcoming that limitation using synthetic data to unbias the training poses, a CNN relative pose regression method approaches the performance of structure-based methods rather than less accurate retrieval approaches, which contradicted the observations reported earlier in the literature. The second question we aimed to answer is *Can a CNN-based pose predictor learn from synthetic views with noise artifacts?*. We showed that even though our synthetic data contains a large number of artifacts (especially in outdoor data), a CNN is capable of learning to disregard those artifacts (without any learned and costly image denoising or refinement modules) during training and improves its performance in the real images.

7.1 Limitations

Domain Adaptation has been widely studied in recent years due to the need of reducing the burden of annotating data. However, DA still needs to tackle some limitations to be functional in real settings. We now expose some specific limitations in our presented approaches (see each individual chapter for

method-specific limitations), as well as general limitations shared by our methods.

Limitations of our Techniques. As a whole, our proposed techniques are constrained by the limited DA setting we considered, namely, single-source domain per task, assumption of source and target datasets representing similar scenes (*e.g.*, a more complicated setting is the adaptation from outdoor to an indoor dataset), and the proposed techniques targeting in some cases narrow settings (*e.g.*, style transfer for multi-style datasets or our projection module for depth completion). Further, although we use pseudo-labels extensively, self-training can degrade the performance when the noise rate is high, which is complicated to evaluate without any target annotation (which is one of the assumptions of Unsupervised Domain Adaptation). Lastly, we assumed weight sharing for both source and target domain, which may not be optimal for a good target domain performance [191].

Limitations of Domain Adaptation. In the DA literature there is a focus on classification tasks while other tasks are still underrepresented. On that note, some of the current methods have limited applications and are tailored to a specific DA setting, which may not work for another variation [225]. There are also some intrinsic limitations to DA, such as required retraining for each new target dataset, which leads to extra complexity compared to other approaches focusing on generalization [79, 352]. Specifically for UDA methods, the collection of an unannotated target dataset can be challenging in some tasks or fields, such as in medical imaging due to strong privacy concerns [181]. In some cases, even the assumption of source data availability can be challenged, *e.g.*, due to privacy concerns [169, 4, 202] or problems in obtaining a large annotated source dataset [366]. Another open problem for UDA is hyperparameter tuning and evaluation [363], as the lack of target annotations requires methods robust to hyperparameter tuning or capable of automatic hyperparameter tuning. Finally, even though Unsupervised Domain Adaptation is a viable solution when target data annotation is not possible, in general, the performance of methods that make use of labeled target data is still better [208, 126, 132]

7.2 Future Directions

Similarly to Deep Learning in general, Domain Adaptation needs better theoretical understanding and guarantees to further improve the current methods. The current DA theory in the Ben-David *et al.* [21,

20] analysis is not followed by the current ad-hoc approaches proposed in state-of-the-art works [3]. A better understanding of the generalization bounds of DA, along with other theoretical insights, can help to improve the performance of DA methods. As for future directions on the techniques applied, we believe that ensuring consistency among multiple tasks can be one of the key ideas to increase DA performance in structured outputs tasks [367, 117]. Image translation will also benefit from ideas capable of addressing some shortcomings, such as generating multiple plausible translations of an input image [59].

Regarding synthetic data, the performance of synthetically-trained models is still limited by the quality of the data. Works aiming to improve synthetic data both in the graphics sense (*e.g.*, illumination, 3D models or textures) [272] and in the sense of high-level scene properties (*e.g.*, number and position of objects) [159, 73] would help to close the domain gap between real and synthetic images. On another note, similarly to our multi-modality based depth estimation (see Chapter 4), leveraging synthetic modalities that show less domain shift than the RGB pixel-space may reduce the visual gap in a synthetic-to-real scenario, *e.g.*, using input depth maps for both synthetic and target real domain to train a semantic segmentation model [380]. We also believe that the data generation pipelines will be directly part of future machine learning models, which may further reduce the domain gap, for example, in [269] the rendering buffers from the simulator are fed into a network for better image style transfer results. Other interesting future directions arising from current works are generating synthetic data directly from the labels [214, 245, 342] or generating synthetically a large amount of both labels and images from a small set of annotated images [369].

Concerning the problem definition, Unsupervised Domain Adaptation is expanding the investigated settings to go further than the single source and single target domain closed-set setting. One specific example is the development of better multi-source approaches [376, 119, 249, 375], which are capable of reusing all of the available source domains for a given task. On that matter, successfully leveraging simultaneously synthetic and real source datasets for training could help the network understand what are the important domain-invariant features for a given task. Going beyond Unsupervised Domain Adaptation, semi-supervised Domain Adaptation, where we leverage a subset of annotated data in the target domain [179], is getting traction as a more realistic short-term middle ground to obtain good performance in the target domain. A relevant trend for the data annotation in the semi-supervised DA

setting is active domain learning, which identifies the target data points that provide the most information when labeled [90, 133, 314] to maximize the performance given fixed annotation budgets, which can reduce the annotation burden. Furthermore, advances in related topics, such as self-supervised learning [47, 120], semi-supervised learning [310, 23] or domain generalization [183, 180] are already being integrated into DA techniques. For example, contrastive approaches, which have been used to obtain high-quality feature extractors in a self-supervised setting [47, 120], are also beneficial for DA settings [150, 156]. Further development in all of these fields will create new techniques and new settings that will help with the data challenge.

7.3 Epilogue

Deep Learning will be the main paradigm in Computer Vision tasks for the near future. However, state-of-the-art pipelines have become bigger and costlier to train [318, 77, 166]. We argue that the machine learning community needs to focus on more accessible and less resource-demanding methods. Recent research has already started to highlight several issues with current Deep Learning pipelines, from the power needed (along with CO_2 emissions contribution) to train these large models, to the limitation that the cost of Deep Learning pipelines put to AI accessible for smaller companies [39]. One of the components making a Deep Learning pipeline expensive is data collection and annotations. Improvements on pretraining or self-training may lead to models that are capable of better generalization, or at least suitable for quick finetuning with few examples, thus reducing the burden of annotation and training. Our direction to reduce the annotation cost was Domain Adaptation, which aims to lower the domain gap between our target domain and existing annotated real or synthetic data. We have investigated the most frequent Domain Adaptation techniques to deal with this domain gap, and explored their applicability to diverse tasks. We then proposed several novel approaches to reduce the distribution shift and to make better use of synthetic data in, semantic, depth, and pose estimation tasks. We believe that further advances in Domain Adaptation will lead to improved results in computationally or data constrained environments.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Alexey Abramov, Christopher Bayer, and Claudio Heller. Keep it simple: Image statistics matching for domain adaptation. *arXiv preprint arXiv:2005.12551*, 2020.

[3] David Acuna, Guojun Zhang, Marc T. Law, and Sanja Fidler. f-domain adversarial learning: Theory and algorithms. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 66–75. PMLR, 18–24 Jul 2021.

[4] Sk Miraj Ahmed, Dripta S Raychaudhuri, Sujoy Paul, Samet Oymak, and Amit K Roy-Chowdhury. Unsupervised multi-source domain adaptation without access to source data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10103–10112, 2021.

[5] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv preprint arXiv:1812.11941*, 2018.

[6] Amjad Almahairi, Sai Rajeshwar, Alessandro Sordoni, Philip Bachman, and Aaron Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 195–204. PMLR, 2018.

[7] Asha Anoosheh, Torsten Sattler, Radu Timofte, Marc Pollefeys, and Luc Van Gool. Night-to-day image translation for retrieval-based localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5958–5964. IEEE, 2019.

[8] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5297–5307, 2016.

[9] Amir Atapour-Abarghouei, Samet Akcay, Gregoire Payen de La Garanderie, and Toby P Breckon. Generative adversarial framework for depth filling via wasserstein metric, cosine transform and domain transfer. *Pattern Recognition*, 91:232–244, 2019.

[10] Amir Atapour-Abarghouei and Toby P Breckon. 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)*, pages 2800–2810, 2018.

[11] Amir Atapour-Abarghouei and Toby P Breckon. 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)*, pages 183–193. IEEE, 2019.

[12] Amir Atapour-Abarghouei and Toby P Breckon. Veritatem dies aperit-temporally consistent depth prediction enabled by a multi-task geometric and semantic scene understanding approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3373–3384, 2019.

[13] Hernan Badino, Daniel Huber, and Takeo Kanade. The CMU Visual Localization Data Set. `http://3dvis.ri.cmu.edu/data-sets/localization`, 2011.

[14] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(12):2481–2495, 2017.

[15] Sungyong Baik, Hyo Jin Kim, Tianwei Shen, Eddy Ilg, Kyoung Mu Lee, and Chris Sweeney. Domain adaptation of learned features for visual localization. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2020.

[16] Mahsa Baktashmotlagh, Mehrtash Harandi, and Mathieu Salzmann. Distribution-matching embedding for visual domain adaptation. *Journal of Machine Learning Research (JMLR)*, 17:1–30, 2016.

[17] Vassileios Balntas, Shuda Li, and Victor Adrian Prisacariu. RelocNet: Continuous metric learning relocalisation using neural nets. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 751–767, 2018.

[18] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (TOG)*, 28(3):24, 2009.

[19] Jonathan T Barron and Ben Poole. The fast bilateral solver. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 617–632. Springer, 2016.

[20] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1):151–175, 2010.

[21] Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. Analysis of representations for domain adaptation. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 19:137, 2007.

[22] Shai Ben-David, Tyler Lu, Teresa Luu, and Dávid Pál. Impossibility theorems for domain adaptation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 129–136, 2010.

[23] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 5049–5059, 2019.

[24] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo-stereo matching with slanted support windows. In *Proceedings of the British Machine Vision Conference (BMVC)*, volume 11, pages 1–11, 2011.

[25] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 343–351, 2016.

[26] Eric Brachmann, Martin Humenberger, Carsten Rother, and Torsten Sattler. On the limits of pseudo ground truth in visual camera re-localisation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6218–6228, 2021.

[27] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - differentiable ransac for camera localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6684–6692, 2017.

[28] Eric Brachmann and Carsten Rother. Learning less is more - 6D camera localization via 3D surface regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4654–4662, 2018.

[29] Eric Brachmann and Carsten Rother. Visual camera re-localization from rgb and rgb-d images using dsac. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 1–1, 2021.

[30] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-aware learning of maps for camera localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2616–2625, 2018.

[31] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Proceedings of*

the Advances in Neural Information Processing Systems (NeurIPS), volume 33, pages 1877–1901, 2020.

[32] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020.

[33] Lucas Caccia, Herke Van Hoof, Aaron Courville, and Joelle Pineau. Deep generative modeling of lidar data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5034–5040. IEEE, 2019.

[34] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1209–1218, 2018.

[35] Qi Cai, Yingwei Pan, Chong-Wah Ngo, Xinmei Tian, Lingyu Duan, and Ting Yao. Exploring object relation in mean teacher for cross-domain detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11457–11466, 2019.

[36] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (6):679–698, 1986.

[37] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–150, 2018.

[38] Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2985–2994, 2019.

[39] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–229. Springer, 2020.

[40] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 8001–8008, 2019.

[41] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. *arXiv preprint arXiv:2102.08604*, 2021.

[42] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7354–7362, 2019.

[43] Wei-Lun Chang, Hui-Po Wang, Wen-Hsiao Peng, and Wei-Chen Chiu. All about structure: Adapting structural information across domains for boosting semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1900–1909, 2019.

[44] Chaoqi Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. Progressive feature alignment for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 627–636, 2019.

[45] Chaoqi Chen, Zebiao Zheng, Xinghao Ding, Yue Huang, and Qi Dou. Harmonizing transferability and discriminability for adapting object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8869–8878, 2020.

[46] Po-Yi Chen, Alexander H Liu, Yen-Cheng Liu, and Yu-Chiang Frank Wang. Towards scene understanding: Unsupervised monocular depth estimation with semantic-aware representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2624–2632, 2019.

[47] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1597–1607. PMLR, 2020.

[48] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 29:730–738, 2016.

[49] Xiaotian Chen, Yuwang Wang, Xuejin Chen, and Wenjun Zeng. S2r-depthnet: Learning a generalizable depth-specific structural representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3034–3043, 2021.

[50] Yuhua Chen, Wen Li, Xiaoran Chen, and Luc Van Gool. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1841–1850, 2019.

[51] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3339–3348, 2018.

[52] Yun Chen, Bin Yang, Ming Liang, and Raquel Urtasun. Learning joint 2d-3d representations for depth completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10023–10032, October 2019.

[53] Yun-Chun Chen, Yen-Yu Lin, Ming-Hsuan Yang, and Jia-Bin Huang. Crdoco: Pixel-level domain transfer with cross-domain consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1791–1800, 2019.

[54] Bin Cheng, Inderjot Singh Saggu, Raunak Shah, Gaurav Bansal, and Dinesh Bharadia. S3net: Semantic-aware self-supervised depth estimation with monocular videos and synthetic data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–69. Springer, 2020.

[55] Xinjing Cheng, Peng Wang, Chenye Guan, and Ruigang Yang. Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 10615–10622, 2020.

[56] Xinjing Cheng, Peng Wang, and Ruigang Yang. Learning depth with convolutional spatial propagation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 42(10):2361–2379, 2019.

[57] Xuelian Cheng, Yiran Zhong, Yuchao Dai, Pan Ji, and Hongdong Li. Noise-aware unsupervised deep lidar-stereo fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6339–6348, 2019.

[58] Boris Chidlovskii and Assem Sadek. Adversarial transfer of pose estimation regression. In *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)*, pages 646–661, 2020.

[59] Eleni Chiou, Eleftheria Panagiotaki, and Iasonas Kokkinos. Beyond deterministic translation for unsupervised domain adaptation. https://elchiou.github.io/UDA_sem_seg/Resources/uda_sem_seg.pdf.

[60] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7781–7790, 2019.

[61] JaeHoon Choi, Dongki Jung, DongHwan Lee, and Changick Kim. Safenet: Self-supervised monocular depth estimation with semantic-aware feature extraction. In *Proceedings of the Advances in Neural Information Processing Systems Workshops (NeurIPSW)*, 2020.

[62] Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Weakly supervised object localization with multi-fold multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(1):189–203, 2017.

[63] Cognilytica. Data preparation labeling for ai 2020. https://www.cognilytica.com/2020/01/31/data-preparation-labeling-for-ai-2020/, 2020. Accessed on 05/10/2021.

[64] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016.

[65] International Data Corporation. Worldwide spending on artificial intelligence is expected to double in four years, reaching $110 billion in 2024, according to new idc spending guide.

https://www.idc.com/getdoc.jsp?containerId=prUS46794720, 2020. Accessed on 05/10/2021.

[66] Corinna Cortes and Vladimir Vapnik. Support vector machine. *Machine learning*, 20(3):273–297, 1995.

[67] Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 3733–3742, 2017.

[68] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(9):1853–1865, 2016.

[69] Adele Cutler, D Richard Cutler, and John R Stevens. Random forests. In *Ensemble Machine Learning*, pages 157–175. Springer, 2012.

[70] Bharath Bhushan Damodaran, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 447–463, 2018.

[71] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[72] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4091–4101, 2021.

[73] Jeevan Devaranjan, Amlan Kar, and Sanja Fidler. Meta-sim2: Unsupervised learning of scene structure for synthetic data generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 715–733. Springer, 2020.

[74] Tom van Dijk and Guido de Croon. How do neural networks see depth in single images? In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2183–2191, 2019.

[75] Mingyu Ding, Zhe Wang, Jiankai Sun, Jianping Shi, and Ping Luo. CamNet: Coarse-to-fine retrieval for camera re-localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2871–2880, 2019.

[76] Anh-Dzung Doan, Yasir Latif, Tat-Jun Chin, Yu Liu, Shin-Fang Ch'ng, Thanh-Toan Do, and Ian Reid. Visual localization under appearance change: filtering approaches. *Neural Computing and Applications*, 33(13):7325–7338, 2021.

[77] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[78] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Proceedings of the Conference on Robot Learning (CoRL)*, pages 1–16, 2017.

[79] Abhimanyu Dubey, Vignesh Ramanathan, Alex Pentland, and Dhruv Mahajan. Adaptive methods for real-world domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14340–14349, 2021.

[80] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2, 2017.

[81] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A trainable cnn for joint detection and description of local features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8084 – 8093, 2019.

[82] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 2366–2374, 2014.

[83] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, 2010.

[84] Jose M Facil, Benjamin Ummenhofer, Huizhong Zhou, Luis Montesano, Thomas Brox, and Javier Civera. Cam-convs: camera-aware multi-scale convolutions for single-view depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11826–11835, 2019.

[85] Zhicheng Fang, Xiaoran Chen, Yuhua Chen, and Luc Van Gool. Towards good practice for cnn-based monocular depth estimation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1091–1100, 2020.

[86] Hao-Zhe Feng, Zhaoyang You, Minghao Chen, Tianye Zhang, Minfeng Zhu, Fei Wu, Chao Wu, and Wei Chen. Kd3a: Unsupervised multi-source decentralized domain adaptation via knowledge distillation. *arXiv preprint arXiv:2011.09757*, 2020.

[87] David Ferstl, Christian Reinbacher, Rene Ranftl, Matthias Rüther, and Horst Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 993–1000, 2013.

[88] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[89] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, number 6, 2018.

[90] Bo Fu, Zhangjie Cao, Jianmin Wang, and Mingsheng Long. Transferable query selection for active domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7272–7281, 2021.

[91] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep Ordinal Regression Network for Monocular Depth Estimation. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 2002–2011, 2018.

[92] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4340–4349, 2016.

[93] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1180–1189, 2015.

[94] Xiao-Shan Gao, Xiao-Rong Hou, Jianlian Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 25(8):930–943, Aug 2003.

[95] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–756, 2016.

[96] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, 2016.

[97] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 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)*, pages 3354–3361. IEEE, 2012.

[98] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 25–38. Springer, 2010.

[99] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pages 248–255, 2019.

[100] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[101] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1440–1448, December 2015.

[102] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[103] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time RGB-D camera relocalization. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 173–179, October 2013.

[104] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3828–3838, 2019.

[105] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 270–279, 2017.

[106] Aaron Gokaslan, Vivek Ramanujan, Daniel Ritchie, Kwang In Kim, and James Tompkin. Improving shape deformation in unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 649–665, 2018.

[107] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073. IEEE, 2012.

[108] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 27:2672–2680, 2014.

[109] Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8977–8986, 2019.

[110] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 529–536, 2004.

[111] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research (JMLR)*, 13(25):723–773, 2012.

[112] Xiao Gu, Yao Guo, Fani Deligianni, and Guang-Zhong Yang. Coupled real-synthetic domain adaptation for real-world deep depth enhancement. *IEEE Transactions on Image Processing*, 29:6343–6356, 2020.

[113] Vitor Guizilini, Rui Hou, Jie Li, Rares Ambrus, and Adrien Gaidon. Semantically-guided representation learning for self-supervised monocular depth. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[114] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 484–500. Springer, 2018.

[115] Xiaoqing Guo, Chen Yang, Baopu Li, and Yixuan Yuan. Metacorrection: Domain-aware meta loss correction for unsupervised domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3927–3936, 2021.

[116] A Gurram, AF Tuna, F Shen, O Urfalioglu, and AM López. Monocular depth estimation through virtual-world supervision and real-world sfm self-supervision. *arXiv preprint arXiv:2103.12209*, 2021.

[117] Emanuela Haller, Elena Burceanu, and Marius Leordeanu. Unsupervised domain adaptation through iterative consensus shift in a multi-task graph. *arXiv preprint arXiv:2103.14417*, 2021.

[118] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 8527–8537, 2018.

[119] Jianzhong He, Xu Jia, Shuaijun Chen, and Jianzhuang Liu. Multi-source domain adaptation with collaborative learning for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11008–11017, 2021.

[120] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020.

[121] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[122] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[123] Lei He, Guanghui Wang, and Zhanyi Hu. Learning depth from single images with deep neural network embedding focal length. *IEEE Transactions on Image Processing*, 27(9):4676–4689, 2018.

[124] Zhenwei He and Lei Zhang. Multi-adversarial faster-rcnn for unrestricted object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6668–6677, 2019.

[125] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(2):328–341, 2007.

[126] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1989–1998, 2018.

[127] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016.

[128] Sara Hooker. The hardware lottery. *arXiv preprint arXiv:2009.06489*, 2020.

[129] Junjie Hu, Yan Zhang, and Takayuki Okatani. Visualization of convolutional neural networks for monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3869–3878, 2019.

[130] Mu Hu, Shuling Wang, Bin Li, Shiyu Ning, Li Fan, and Xiaojin Gong. Penet: Towards precise and efficient image guided depth completion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 13656–13662, 2021.

[131] Yuan-Ting Hu, Jiahong Wang, Raymond A Yeh, and Alexander G Schwing. Sail-vos 3d: A synthetic dataset and baselines for object detection and 3d mesh reconstruction from video data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3364–3374, 2021.

[132] Jiaxing Huang, Dayan Guan, Aoran Xiao, and Shijian Lu. Cross-view regularization for domain adaptive panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10133–10144, 2021.

[133] Sheng-Jun Huang, Jia-Wei Zhao, and Zhao-Yang Liu. Cost-effective training of deep cnns with active model adaptation. In *Proceedings of the ACM International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, pages 1580–1588, 2018.

[134] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1501–1510, 2017.

[135] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–189, 2018.

[136] Yu-Kai Huang, Tsung-Han Wu, Yueh-Cheng Liu, and Winston H Hsu. Indoor depth completion with boundary consistency and self-attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 1070–1078, 2019.

[137] Zixuan Huang, Junming Fan, Shuai Yi, Xiaogang Wang, and Hongsheng Li. Hms-net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion. *arXiv preprint arXiv:1808.08685*, 2018.

[138] Martin Humenberger, Yohann Cabon, Nicolas Guerin, Julien Morat, Jérôme Revaud, Philippe Rerole, Noé Pion, Cesar de Souza, Vincent Leroy, and Gabriela Csurka. Robust image retrieval-based visual localization using kapture. *arXiv preprint arXiv:2007.13867*, 2020.

[139] Braden Hurl, Krzysztof Czarnecki, and Steven Waslander. Precise synthetic image and lidar (presil) dataset for autonomous vehicle perception. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 2522–2529. IEEE, 2019.

[140] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5001–5009, 2018.

[141] Takashi Isobe, Xu Jia, Shuaijun Chen, Jianzhong He, Yongjie Shi, Jianzhuang Liu, Huchuan Lu, and Shengjin Wang. Multi-target domain adaptation with collaborative consistency learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8187–8196, 2021.

[142] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1125–1134, 2017.

[143] Thorbjørn Mosekjær Iversen and Dirk Kraft. Generation of synthetic kinect depth images based on empirical noise model. *Electronics Letters*, 53(13):856–858, 2017.

[144] Ara Jafarzadeh, Manuel López Antequera, Pau Gargallo, Yubin Kuang, Carl Toft, Fredrik Kahl, and Torsten Sattler. Crowddriven: A new challenging dataset for outdoor visual localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9845–9855, 2021.

[145] Maximilian Jaritz, Raoul De Charette, Emilie Wirbel, Xavier Perrotton, and Fawzi Nashashibi. Sparse and dense data with cnns: Depth completion and semantic segmentation. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 52–60. IEEE, 2018.

[146] Jianbo Jiao, Ying Cao, Yibing Song, and Rynson Lau. Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss. In *Proceedings of the Euro-*

pean Conference on Computer Vision (ECCV), pages 53–69, 2018.

[147] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *International Journal of Computer Vision (IJCV)*, 129(2):517–547, 2021.

[148] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 694–711. Springer, 2016.

[149] M. Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2017.

[150] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4893–4902, 2019.

[151] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. Meta-sim: Learning to generate synthetic datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4551–4560, 2019.

[152] Alex Kendall and Roberto Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4762–4769, 2016.

[153] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5974–5983, 2017.

[154] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2938–2946, 2015.

[155] Mehran Khodabandeh, Arash Vahdat, Mani Ranjbar, and William G Macready. A robust learning approach to domain adaptive object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 480–490, 2019.

[156] Donghyun Kim, Kuniaki Saito, Tae-Hyun Oh, Bryan A Plummer, Stan Sclaroff, and Kate Saenko. Cross-domain self-supervised learning for domain adaptation with few source labels. *arXiv preprint arXiv:2003.08264*, 2020.

[157] Myeongjin Kim and Hyeran Byun. Learning texture invariant representation for domain adaptation of semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12975–12984, 2020.

[158] Seunghyeon Kim, Jaehoon Choi, Taekyung Kim, and Changick Kim. Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6092–6101, 2019.

[159] Seung Wook Kim, Jonah Philion, Antonio Torralba, and Sanja Fidler. Drivegan: Towards a controllable high-quality neural simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5820–5829, 2021.

[160] Seung Wook Kim, Yuhao Zhou, Jonah Philion, Antonio Torralba, and Sanja Fidler. Learning to simulate dynamic environments with gamegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1231–1240, 2020.

[161] Taekyung Kim, Minki Jeong, Seunghyeon Kim, Seokeon Choi, and Changick Kim. Diversify and match: A domain adaptive representation learning paradigm for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12456–12465, 2019.

[162] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[163] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollar. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6399–6408, 2019.

[164] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9404–9413, 2019.

[165] Marvin Klingner, Jan-Aike Termöhlen, Jonas Mikolajczyk, and Tim Fingscheidt. Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 582–600. Springer, 2020.

[166] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 491–507. Springer, 2020.

[167] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 1097–1105, 2012.

[168] Jogendra Nath Kundu, Nishank Lakkakula, and R Venkatesh Babu. Um-adapt: Unsupervised multi-task adaptation using adversarial cross-task distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1436–1445, 2019.

[169] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4544–4553, 2020.

[170] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 239–248. IEEE, 2016.

[171] John Lambert, Liu Zhuang, Ozan Sener, James Hays, and Vladlen Koltun. MSeg: A composite dataset for multi-domain semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2879–2888, 2020.

[172] Michael J Landau, Benjamin Y Choo, and Peter A Beling. Simulating kinect infrared and depth images. *IEEE Transactions on Cybernetics*, 46(12):3018–3031, 2015.

[173] Nalpantidis Lazaros, Georgios Christou Sirakoulis, and Antonios Gasteratos. Review of stereo vision algorithms: from software to hardware. *International Journal of Optomechatronics*, 2(4):435–462, 2008.

[174] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[175] Byeong-Uk Lee, Hae-Gon Jeon, Sunghoon Im, and In So Kweon. Depth completion with deep geometry and context guidance. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3281–3287. IEEE, 2019.

[176] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10285–10295, 2019.

[177] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Proceedings of the International Conference on Machine Learning Workshops (ICMLW)*, volume 3, page 2, 2013.

[178] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*, 2019.

[179] Bo Li, Yezhen Wang, Shanghang Zhang, Dongsheng Li, Kurt Keutzer, Trevor Darrell, and Han Zhao. Learning invariant representations and risks for semi-supervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1104–1113, 2021.

[180] Daiqing Li, Junlin Yang, Karsten Kreis, Antonio Torralba, and Sanja Fidler. Semantic segmentation with generative models: Semi-supervised learning and strong out-of-domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8300–8311, 2021.

[181] Haoliang Li, Yufei Wang, Renjie Wan, Shiqi Wang, Tie-Qiang Li, and Alex Kot. Domain generalization for medical imaging classification with linear-dependency regularization. In

Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), volume 33, pages 3118–3129, 2020.

[182] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Learning to learn from noisy labeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5051–5059, 2019.

[183] Lei Li, Ke Gao, Juan Cao, Ziyao Huang, Yepeng Weng, Xiaoyue Mi, Zhengze Yu, Xiaoya Li, and Boyang Xia. Progressive domain expansion network for single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 224–233, 2021.

[184] Rui Li, Qing Mao, Pei Wang, Xiantuo He, Yu Zhu, Jinqiu Sun, and Yanning Zhang. Semantic-guided representation enhancement for self-supervised monocular trained depth estimation. *arXiv preprint arXiv:2012.08048*, 2020.

[185] Wenbin Li, Sajad Saeedi, John McCormac, Ronald Clark, Dimos Tzoumanikas, Qing Ye, Yuzhong Huang, Rui Tang, and Stefan Leutenegger. Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. In *Proceedings of the British Machine Vision Conference (BMVC)*, page 77, 2018.

[186] Xiaotian Li, Shuzhe Wang, Yi Zhao, Jakob Verbeek, and Juho Kannala. Hierarchical scene coordinate classification and regression for visual localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11983–11992, 2020.

[187] Xiaotian Li, Juha Ylioinas, and Juho Kannala. Full-frame scene coordinate regression for image-based localization. In *Proceedings of the Robotics: Science and Systems Conference (RSS)*. University of Queensland, 2018.

[188] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 453–468, 2018.

[189] Yanghao Li, Naiyan Wang, Jianping Shi, Xiaodi Hou, and Jiaying Liu. Adaptive batch normalization for practical domain adaptation. *Pattern Recognition*, 80:109–117, 2018.

[190] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. In *Proceedings of the International Conference on Learning Representations Workshops (ICLRW)*, 2017.

[191] Yunsheng Li, Lu Yuan, Yinpeng Chen, Pei Wang, and Nuno Vasconcelos. Dynamic transfer for multi-source domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10998–11007, 2021.

[192] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2041–2050, 2018.

[193] Qing Lian, Fengmao Lv, Lixin Duan, and Boqing Gong. Constructing self-motivated pyramid curriculums for cross-domain semantic segmentation: A non-adversarial approach. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6758–6767, 2019.

[194] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[195] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014.

[196] Yuval Litvak, Armin Biess, and Aharon Bar-Hillel. Learning pose estimation for high-precision robotic assembly using simulated depth images. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3521–3527. IEEE, 2019.

[197] Keng-Chi Liu, Yi-Ting Shen, Jan P Klopp, and Liang-Gee Chen. What synthesis is missing: Depth adaptation integrated with weak supervision for indoor scene parsing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7345–7354, 2019.

[198] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Proceedings of the Advances in Neural Information Processing Systems*

(NeurIPS), 33, 2020.

[199] Lina Liu, Xibin Song, Xiaoyang Lyu, Junwei Diao, Mengmeng Wang, Yong Liu, and Liangjun Zhang. Fcfr-net: Feature fusion based coarse-to-fine residual learning for monocular depth completion. *arXiv preprint arXiv:2012.08270*, 2020.

[200] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 1519–1529, 2017.

[201] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 21–37. Springer, 2016.

[202] Yuang Liu, Wei Zhang, and Jun Wang. Source-free domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1215–1224, 2021.

[203] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 97–105. PMLR, 2015.

[204] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 1647–1657, 2018.

[205] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer joint matching for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1410–1417, 2014.

[206] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 29:136–144, 2016.

[207] Adrian Lopez-Rodriguez, Benjamin Busam, and Krystian Mikolajczyk. Project to adapt: Domain adaptation for depth completion from noisy and sparse sensor data. In Hiroshi Ishikawa, Cheng-Lin Liu, Tomás Pajdla, and Jianbo Shi, editors, *Proceedings of the Asian Conference on*

Computer Vision (ACCV), volume 12622 of *Lecture Notes in Computer Science*, pages 330–348. Springer, 2020.

[208] Adrian Lopez-Rodriguez and Krystian Mikolajczyk. Domain adaptation for object detection via style consistency. In *Proceedings of the British Machine Vision Conference (BMVC)*, page 232, 2019.

[209] Adrian Lopez-Rodriguez and Krystian Mikolajczyk. Desc: Domain adaptation for depth estimation via semantic consistency. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2020.

[210] David G. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision (IJCV)*, pages 91–110, 2004.

[211] Fangchang Ma, Guilherme Venturelli Cavalheiro, and Sertac Karaman. Self-supervised sparse-to-dense: self-supervised depth completion from lidar and monocular camera. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3288–3295. IEEE, 2019.

[212] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017.

[213] Fangchang Mal and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8. IEEE, 2018.

[214] Arun Mallya, Ting-Chun Wang, Karan Sapra, and Ming-Yu Liu. World-consistent video-to-video synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 359–378. Springer, 2020.

[215] Massimiliano Mancini, Lorenzo Porzi, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Boosting domain adaptation by discovering latent domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3771–3780, 2018.

[216] Sivabalan Manivasagam, Shenlong Wang, Kelvin Wong, Wenyuan Zeng, Mikita Sazanovich, Shuhan Tan, Bin Yang, Wei-Chiu Ma, and Raquel Urtasun. Lidarsim: Realistic lidar simulation

by leveraging the real world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11167–11176, 2020.

[217] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Proceedings of the Conference on Learning Theory (COLT)*, 2009.

[218] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2794–2802, 2017.

[219] Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. Autodial: Automatic domain alignment layers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5067–5075, 2017.

[220] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7210–7219, 2021.

[221] Luke Melas-Kyriazi and Arjun K Manrai. Pixmatch: Unsupervised domain adaptation via pixelwise consistency training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12435–12445, 2021.

[222] Yue Meng, Yongxi Lu, Aman Raj, Samuel Sunarjo, Rui Guo, Tara Javidi, Gaurav Bansal, and Dinesh Bharadia. Signet: Semantic instance aided unsupervised 3d geometry perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9810–9820, 2019.

[223] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3061–3070, 2015.

[224] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 405–421, 2020.

[225] Yu Mitsuzumi, Go Irie, Daiki Ikami, and Takashi Shibata. Generalized domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1084–1093, 2021.

[226] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 41(8):1979–1993, 2018.

[227] Rohit Mohan and Abhinav Valada. Efficientps: Efficient panoptic segmentation. *International Journal of Computer Vision (IJCV)*, 129(5):1551–1579, 2021.

[228] Arsalan Mousavian, Hamed Pirsiavash, and Jana Košecká. Joint semantic segmentation and depth estimation with deep convolutional networks. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 611–619. IEEE, 2016.

[229] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 10–18. PMLR, 2013.

[230] Markus S Mueller, Torsten Sattler, Marc Pollefeys, and Boris Jutzi. Image-to-image translation for enhanced feature matching, image retrieval and visual localization. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 4:111–119, 2019.

[231] Kazuto Nakashima and Ryo Kurazume. Learning to drop points for lidar scan synthesis. *arXiv preprint arXiv:2102.11952*, 2021.

[232] Tayyab Naseer and Wolfram Burgard. Deep regression for monocular camera-based 6-DoF global localization in outdoor environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1525–1530, 2017.

[233] Jogendra Nath Kundu, Phani Krishna Uppala, Anuj Pahuja, and R Venkatesh Babu. Adadepth: Unsupervised content congruent adaptation for depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2656–2665, 2018.

[234] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Proceedings of the Advances in Neural Information Processing Systems Workshops (NeurIPSW)*, 2011.

[235] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4990–4999, 2017.

[236] Tony Ng, Vassileios Balntas, Yurun Tian, and Krystian Mikolajczyk. SOLAR: Second-order loss and attention for image retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 253–270, 2020.

[237] Tony Ng, Adrian Lopez-Rodriguez, Vassileios Balntas, and Krystian Mikolajczyk. Reassessing the limitations of cnn methods for camera pose regression. *arXiv preprint arXiv:2108.07260*, 2021.

[238] Kiri Nichol. Painter by numbers. `https://www.kaggle.com/c/painter-by-numbers`, 2016. Accessed on 05/10/2021.

[239] Yusuke Niitani, Toru Ogawa, Shunta Saito, and Masaki Saito. Chainercv: a library for deep learning in computer vision. In *Proceedings of the ACM International Conference on Multimedia (MM)*, pages 1217–1220. ACM, 2017.

[240] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. In *ACM Transactions on Graphics (TOG)*, volume 38, pages 1–15, 2019.

[241] NVIDIA. What is synthetic data? `https://blogs.nvidia.com/blog/2021/06/08/what-is-synthetic-data/`, 2021. Accessed on 05/10/2021.

[242] Fei Pan, Inkyu Shin, Francois Rameau, Seokju Lee, and In So Kweon. Unsupervised intra-domain adaptation for semantic segmentation through self-supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3764–3773, 2020.

[243] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 754–763, 2017.

[244] Jinsun Park, Kyungdon Joo, Zhe Hu, Chi-Kuei Liu, and In-So Kweon. Non-local spatial propagation network for depth completion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 120–136, 2020.

[245] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2337–2346, 2019.

[246] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Autodiff Workshop*, 2017.

[247] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035. 2019.

[248] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3934–3941, 2018.

[249] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1406–1415, 2019.

[250] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 5102–5112. PMLR, 2019.

[251] Xingchao Peng, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 2021–2026, 2018.

[252] Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. *arXiv preprint arXiv:1806.09755*, 2018.

[253] Andrea Pilzer, Dan Xu, Mihai Puscas, Elisa Ricci, and Nicu Sebe. Unsupervised adversarial depth estimation using cycled generative networks. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 587–595. IEEE, 2018.

[254] Noé Pion, Martin Humenberger, Gabriela Csurka, Yohann Cabon, and Torsten Sattler. Benchmarking image retrieval for visual localization. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 483–494, 2020.

[255] Benjamin Planche, Ziyan Wu, Kai Ma, Shanhui Sun, Stefan Kluckner, Oliver Lehmann, Terrence Chen, Andreas Hutter, Sergey Zakharov, Harald Kosch, et al. Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5 d recognition. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 1–10. IEEE, 2017.

[256] Koutilya PNVR, Hao Zhou, and David Jacobs. Sharingan: Combining synthetic and real data for unsupervised geometry estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13971–13980, June 2020.

[257] Horia Porav, Will Maddern, and Paul Newman. Adversarial training for adverse conditions: Robust metric localisation using appearance transfer. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1011–1018. IEEE, 2018.

[258] Jiaxiong Qiu, Zhaopeng Cui, Yinda Zhang, Xingdi Zhang, Shuaicheng Liu, Bing Zeng, and Marc Pollefeys. 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)*, pages 3313–3322, 2019.

[259] Filip Radenović, Giorgos Tolias, and Ondřej Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–20, 2016.

[260] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning CNN image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 1655–1668, 2018.

[261] Leinar Ramos and Jitendra Subramanyam. Maverick research: Forget about your real data – synthetic data is the future of ai. *Gartner*, 2021.

[262] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 1–1, 2020.

[263] Ievgen Redko, Emilie Morvant, Amaury Habrard, Marc Sebban, and Younès Bennani. A survey on domain adaptation theory: learning bounds and theoretical guarantees. *arXiv preprint arXiv:2004.11829*, 2020.

[264] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[265] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2015.

[266] Jerome Revaud, Jon Almazán, Rafael Sampaio de Rezende, and César Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5106–5115, 2019.

[267] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2D2: Repeatable and reliable detector and descriptor. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 12405–12415, 2019.

[268] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3674–3683, 2020.

[269] Stephan R Richter, Hassan Abu AlHaija, and Vladlen Koltun. Enhancing photorealism enhancement. *arXiv preprint arXiv:2105.04619*, 2021.

[270] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 12364, pages 623–640, 2020.

[271] Daniel Ritchie, Kai Wang, and Yu-an Lin. Fast and flexible indoor scene synthesis via deep convolutional generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6182–6190, 2019.

[272] Mike Roberts and Nathan Paczan. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. *arXiv preprint arXiv:2011.02523*, 2020.

[273] Eduardo Romera, Luis M Bergasa, Kailun Yang, Jose M Alvarez, and Rafael Barea. Bridging the day and night domain gap for semantic segmentation. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 1312–1318. IEEE, 2019.

[274] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241, 2015.

[275] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. 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)*, pages 3234–3243, 2016.

[276] Subhankar Roy, Aliaksandr Siarohin, Enver Sangineto, Samuel Rota Bulo, Nicu Sebe, and Elisa Ricci. Unsupervised domain adaptation using feature-whitening and consensus loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9471–9480, 2019.

[277] Aruni RoyChowdhury, Prithvijit Chakrabarty, Ashish Singh, SouYoung Jin, Huaizu Jiang, Liangliang Cao, and Erik Learned-Miller. Automatic adaptation of object detectors to new domains using self-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 780–790, 2019.

[278] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 41(4):801–814, 2018.

[279] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Residual parameter transfer for deep domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4339–4348, 2018.

[280] Faraz Saeedan and Stefan Roth. Boosting monocular depth with panoptic segmentation maps. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3853–3862, 2021.

[281] Aleksandr Safin, Maxim Kan, Nikita Drobyshev, Oleg Voynov, Alexey Artemov, Alexander Filippov, Denis Zorin, and Evgeny Burnaev. Towards unpaired depth enhancement and super-resolution in the wild. *arXiv preprint arXiv:2105.12038*, 2021.

[282] Suman Saha, Anton Obukhov, Danda Pani Paudel, Menelaos Kanakis, Yuhua Chen, Stamatios Georgoulis, and Luc Van Gool. Learning to relate depth and semantics for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8197–8207, 2021.

[283] Soham Saha, Girish Varma, and C. V. Jawahar. Improved visual relocalization by discovering anchor points. In *Proceedings of the British Machine Vision Conference (BMVC)*, page 164, 2018.

[284] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. Universal domain adaptation through self supervision. *arXiv preprint arXiv:2002.07953*, 2020.

[285] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 2988–2997, 2017.

[286] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Strong-weak distribution alignment for adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6956–6965, 2019.

[287] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 153–168, 2018.

[288] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision (IJCV)*, 126(9):973–992, Sep 2018.

[289] Hamed Sarbolandi, Damien Lefloch, and Andreas Kolb. Kinect range sensing: Structured-light versus time-of-flight kinect. *Computer Vision and Image Understanding*, 139:1–20, 2015.

[290] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12716–12725, 2019.

[291] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4937–4946, 2020.

[292] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(9):1744–1756, 2017.

[293] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6dof outdoor visual localization in changing conditions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8601–8610, 2018.

[294] Torsten Sattler, Akihiko Torii, Josef Sivic, Marc Pollefeys, Hajime Taira, Masatoshi Okutomi, and Tomas Pajdla. Are large-scale 3d models really necessary for accurate visual localization? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 814–829, 2017.

[295] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image retrieval for image-based localization revisited. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 1–12, 2012.

[296] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura-Taixe Leal. Understanding the limitations of CNN-based absolute camera pose regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3302–3312, 2019.

[297] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9338–9346, 2019.

[298] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 31(5):824–840, 2008.

[299] Scale AI. Pandaset. https://scale.com/open-datasets/pandaset, 2020. Accessed on 05/10/2021.

[300] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, 47(1-3):7–42, 2002.

[301] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 501–518. Springer, 2016.

[302] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.

[303] Lama Seoud, Jonathan Boisvert, Marc-Antoine Drouin, Michel Picard, and Guy Godin. Increasing the robustness of cnn-based human body segmentation in range images by modeling sensor-specific artifacts. In *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)*, pages 729–743, 2018.

[304] Yuhu Shan, Wen Feng Lu, and Chee Meng Chew. Pixel and feature level based domain adaptation for object detection in autonomous driving. *Neurocomputing*, 367:31–38, 2019.

[305] Tianwei Shen, Lei Zhou, Zixin Luo, Yao Yao, Shiwei Li, Jiahui Zhang, Tian Fang, and Long Quan. Self-supervised learning of depth and motion under photometric inconsistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 4044–4053, 2019.

[306] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2107–2116, 2017.

[307] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 746–760. Springer, 2012.

[308] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[309] Saurabh Singh and Abhinav Shrivastava. Evalnorm: Estimating batch normalization statistics for evaluation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3633–3641, 2019.

[310] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 596–608, 2020.

[311] Xiao Song, Guorun Yang, Xinge Zhu, Hui Zhou, Zhe Wang, and Jianping Shi. Adastereo: a simple and efficient approach for adaptive stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10328–10337, 2021.

[312] Pratul P. Srinivasan, Richard Tucker, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 175–184, 2019.

[313] Kartik Srivastava, Akash Kumar Singh, and Guruprasad M Hegde. Multi modal semantic segmentation using synthetic data. *arXiv preprint arXiv:1910.13676*, 2019.

[314] Jong-Chyi Su, Yi-Hsuan Tsai, Kihyuk Sohn, Buyu Liu, Subhransu Maji, and Manmohan Chandraker. Active adversarial domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 739–748, 2020.

[315] Peng Su, Kun Wang, Xingyu Zeng, Shixiang Tang, Dapeng Chen, Di Qiu, and Xiaogang Wang. Adapting object detectors with conditional domain normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 403–419. Springer, 2020.

[316] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 30, pages 2058–2065, 2016.

[317] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 443–450. Springer, 2016.

[318] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 843–852, 2017.

[319] Chris Sweeney, Greg Izatt, and Russ Tedrake. A supervised approach to predicting noise in depth images. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 796–802. IEEE, 2019.

[320] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. *arXiv preprint arXiv:2106.14405*, 2021.

[321] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. Inloc: Indoor visual localization with dense matching and

view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7199–7209, 2018.

[322] Kevin Tang, Vignesh Ramanathan, Li Fei-Fei, and Daphne Koller. Shifting weights: Adapting object detectors from image to video. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 638–646, 2012.

[323] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 1195–1204, 2017.

[324] Yurun Tian, Axel Barroso-Laguna, Tony Ng, Vassileios Balntas, and Krystian Mikolajczyk. HyNet: Learning local descriptor with hybrid similarity measure and triplet loss. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[325] Beau Tippetts, Dah Jye Lee, Kirt Lillywhite, and James Archibald. Review of stereo vision algorithms and their suitability for resource-limited systems. *Journal of Real-Time Image Processing (JRTIP)*, 11(1):5–25, 2016.

[326] Akihiko Torii, Relja Arandjelović, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1808–1817, 2015.

[327] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1521–1528. IEEE, 2011.

[328] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7472–7481, 2018.

[329] Eric Tzeng, Kaylee Burns, Kate Saenko, and Trevor Darrell. Splat: Semantic pixel-level adaptation transforms for detection. *arXiv preprint arXiv:1812.00929*, 2018.

[330] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2962–2971, 2017.

[331] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 11–20. IEEE, 2017.

[332] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 48, pages 1349–1357, 2016.

[333] Wouter Van Gansbeke, Davy Neven, Bert De Brabandere, and Luc Van Gool. Sparse and noisy lidar completion with rgb guidance and uncertainty. In *Proceedings of the International Conference on Machine Vision Applications (MVA)*, pages 1–6. IEEE, 2019.

[334] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[335] Riccardo Volpi, Pietro Morerio, Silvio Savarese, and Vittorio Murino. Adversarial feature augmentation for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5495–5504, 2018.

[336] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2517–2526, 2019.

[337] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Dada: Depth-aware domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7364–7373, 2019.

[338] Florian Walch, Caner Hazirbas, Laura Leal-Taixé, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using LSTMs for structured feature correlation. In

Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 627–637, 2017.

[339] Yiming Wan, Wei Gao, Sheng Han, and Yihong Wu. Boosting image-based localization via randomly geometric data augmentation. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 688–692. IEEE, 2020.

[340] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. Atloc: Attention guided camera localization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 10393–10401, 2020.

[341] Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Transferable normalization: Towards improving transferability of deep neural networks. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 32:1953–1963, 2019.

[342] Yi Wang, Lu Qi, Ying-Cong Chen, Xiangyu Zhang, and Jiaya Jia. Image synthesis via semantic composition. *arXiv preprint arXiv:2109.07053*, 2021.

[343] Alex Wong, Xiaohan Fei, Stephanie Tsuei, and Stefano Soatto. Unsupervised depth completion from visual inertial odometry. *IEEE Robotics and Automation Letters*, 5(2):1899–1906, 2020.

[344] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893. IEEE, 2018.

[345] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[346] Zuxuan Wu, Xintong Han, Yen-Liang Lin, Mustafa Gokhan Uzunbas, Tom Goldstein, Ser Nam Lim, and Larry S Davis. Dcan: Dual channel-wise alignment networks for unsupervised scene adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 518–534, 2018.

[347] Wangmeng Xiang, Hongwei Yong, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Second-order camera-aware color transformation for cross-domain person re-identification. In

Proceedings of the Asian Conference on Computer Vision (ACCV), volume 12623, pages 36–53, 2020.

[348] Rongchang Xie, Fei Yu, Jiachao Wang, Yizhou Wang, and Li Zhang. Multi-level domain adaptive learning for cross-domain detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 3213–3219. IEEE Computer Society, 2019.

[349] Li Xin. Statistics of acceptance rate for the main ai conferences. `https://github.com/lixin4ever/Conference-Acceptance-Rate`, 2021.

[350] Xin Xiong, Haipeng Xiong, Ke Xian, Chen Zhao, Zhiguo Cao, and Xin Li. Sparse-to-dense depth completion revisited: Sampling strategy and graph construction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 12366, pages 682–699, 2020.

[351] Chang-Dong Xu, Xing-Ran Zhao, Xin Jin, and Xiu-Shen Wei. Exploring categorical regularization for domain adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11724–11733, 2020.

[352] Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A fourier-based framework for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14383–14392, 2021.

[353] Yan Xu, Xinge Zhu, Jianping Shi, Guofeng Zhang, Hujun Bao, and Hongsheng Li. Depth completion from sparse lidar data with depth-normal constraints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2811–2820, 2019.

[354] Fei Xue, Xin Wu, Shaojun Cai, and Junqin Wang. Learning multi-view camera relocalization with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11372–11381, 2020.

[355] Nan Yang, Rui Wang, Jorg Stuckler, and Daniel Cremers. Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 817–833. Springer, 2018.

[356] Yanchao Yang, Yuefan Shen, Youyi Zheng, C Karen Liu, and Leonidas Guibas. Dcl: Differential contrastive learning for geometry-aware depth synthesis. *arXiv preprint arXiv:2107.13087*, 2021.

[357] Yanchao Yang, Alex Wong, and Stefano Soatto. Dense depth posterior (ddp) from single image and sparse range. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3353–3362, 2019.

[358] Keren Ye, Mingda Zhang, Adriana Kovashka, Wei Li, Danfeng Qin, and Jesse Berent. Cap2det: Learning to amplify weak caption supervision for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9686–9695, 2019.

[359] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1983–1992, 2018.

[360] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. *arXiv preprint arXiv:1903.09760*, 2019.

[361] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 27:3320–3328, 2014.

[362] Kaichao You, Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Universal domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2720–2729, 2019.

[363] Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. Towards accurate model selection in deep unsupervised domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 7124–7133. PMLR, 2019.

[364] Xiangyu Yue, Bichen Wu, Sanjit A Seshia, Kurt Keutzer, and Alberto L Sangiovanni-Vincentelli. A lidar point cloud generator: from a virtual world to autonomous driving. In *Proceedings of the ACM International Conference on Multimedia Retrieval (ICMR)*, pages 458–464. ACM, 2018.

[365] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2100–2110, 2019.

[366] Xiangyu Yue, Zangwei Zheng, Shanghang Zhang, Yang Gao, Trevor Darrell, Kurt Keutzer, and Alberto Sangiovanni Vincentelli. Prototypical cross-domain self-supervised learning for few-shot unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13834–13844, 2021.

[367] Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11197–11206, 2020.

[368] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3801–3809, 2018.

[369] Yuxuan Zhang, Huan Ling, Jun Gao, Kangxue Yin, Jean-Francois Lafleche, Adela Barriuso, Antonio Torralba, and Sanja Fidler. Datasetgan: Efficient labeled data factory with minimal human effort. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10145–10155, 2021.

[370] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 7404–7413. PMLR, 2019.

[371] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Zequn Jie, Xiang Li, and Jian Yang. Joint task-recursive learning for semantic segmentation and depth estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 235–251, 2018.

[372] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 8778–8788, 2018.

[373] Zichao Zhang, Torsten Sattler, and Davide Scaramuzza. Reference pose generation for long-term visual localization via learned features and view synthesis. *International Journal of Computer Vision (IJCV)*, 129(4):821–844, Dec 2020.

[374] Shanshan Zhao, Huan Fu, Mingming Gong, and Dacheng Tao. Geometry-aware symmetric domain adaptation for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9788–9798, 2019.

[375] Sicheng Zhao, Bo Li, Pengfei Xu, and Kurt Keutzer. Multi-source domain adaptation in the deep learning era: A systematic survey. *arXiv preprint arXiv:2002.12169*, 2020.

[376] Sicheng Zhao, Bo Li, Xiangyu Yue, Yang Gu, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. Multi-source domain adaptation for semantic segmentation. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 7285–7298, 2019.

[377] Yunhan Zhao, Shu Kong, and Charless Fowlkes. Camera pose matters: Improving depth prediction by mitigating pose distribution bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15759–15768, 2021.

[378] Yunhan Zhao, Shu Kong, Daeyun Shin, and Charless Fowlkes. Domain decluttering: Simplifying images to mitigate synthetic-real domain shift and improve depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3330–3340, 2020.

[379] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783. Springer, 2018.

[380] Brady Zhou, Nimit Kalra, and Philipp Krähenbühl. Domain adaptation through task distillation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 664–680. Springer, 2020.

[381] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1851–1858, 2017.

[382] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2223–2232, 2017.

[383] Shengjie Zhu, Garrick Brazil, and Xiaoming Liu. The edge of depth: Explicit constraints between segmentation and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13116–13125, 2020.

[384] Xinge Zhu, Jiangmiao Pang, Ceyuan Yang, Jianping Shi, and Dahua Lin. Adapting object detectors via selective cross-domain alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 687–696, 2019.

[385] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5982–5991, 2019.

[386] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 289–305, 2018.

[387] Laurent Zwald and Sophie Lambert-Lacroix. The berhu penalty and the grouped effect. *arXiv preprint arXiv:1207.6868*, 2012.