2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Test-Time Synthetic-to-Real Adaptive Depth Estimation

Eojindl Yi[1] and Junmo Kim[1]

***Abstract*— Is it possible for a synthetic to realistic domain adapted neural network in single image depth estimation to truly generalize on real world data? The resultant, adapted model will only generalize on the realistic domain dataset, which only reflects a small portion of the true, *real world*. As a result, the network still has to cope with the potential danger of domain shift between the realistic domain dataset and the real world data. Instead, a viable solution is to design the model to be capable of continuously adapting to the distribution of data it receives at test-time. In this paper, we propose a depth estimation method that is capable of adapting to the domain shift at test-time. Our method adapts to the unseen test-time domain, by updating the network using our proposed objective functions. Following former work, we reduce the entropy of the current prediction for refinement and adaptation. We propose a *Logit Order Enforcement* loss that can prevent the network from deviating into wrong solutions, which can result from the mere reduction of the aforementioned entropy. Qualitative and quantitative results show the effectiveness of our method. Our method reduces the dependency on training data by $5.8\times$ on average, while achieving comparable performance to state-of-the-art unsupervised domain adaptation (UDA) and domain generalization methods (DG) on the KITTI dataset.**

I. Introduction

Depth estimation is a fundamental task in computer vision, which has many connections to diverse research divisions such as robot navigation, autonomous driving and scene understanding. Traditional research has focused on estimating depth from a pair of stereo images. More recently, deep neural network based methods have emerged, which are trained to infer depth from a single image. Scale ambiguity is an inherent issue of single image-based methods, but it is expected that the network will cope against such issues by learning how to reason from image context.

As a natural consequence, the success of deep learning based depth estimation methods relies on the quality and quantity of training data. However, accurately annotated depth data cannot be obtained unless multiple camera pairs or LiDAR sensors are prepared at the time of data acquisition, which makes it costly to obtain data. To resolve the shortcoming of data, strategies such as unsupervised domain adaptation (UDA) [1], [2], [3] or mixing datasets [4], [5] have been attempted. UDA attempts to transfer the learned knowledge from a labeled source domain to a target domain with unlabeled data. Usually, the source domain is set as the synthetic domain, because datasets from the synthetic domain are easy to obtain. Dataset mixing has been attempted to learn from a multiple genera of data, in order to increase generalization. However, the datasets all differ in format (disparity, relative depth or metric depth) and scale and shift. Therefore, a network is trained to predict depth up to an unknown scale and shift, which means its usage is only limited to certain applications.

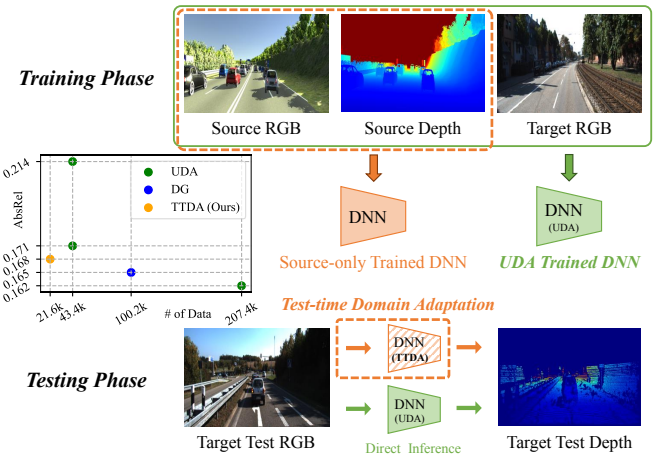

Fig. 1. **Comparison of our *Test-time Domain Adaptation* setup against the *Unsupervised Domain Adaptation* setup.** Our setup does not have access to the target data during training, but updates the network during the testing phase to adapt to the domain shift. This lessens the dependence on the amount of data required for domain adaptation, denoted by the low *# of Data* on the left figure. Still, our method achieves a low value of *Abs Rel* which is comparable to other methods which use more data. The methods which correspond to each dot in the figure can be found in Tab. I.

Regardless of the amount of data we have at hand, the data can never cover the distribution of the real world, which implies that our trained model will always be at the danger of suffering from low performance due to domain or distribution shift at test-time. To overcome such a problem, recent advances in the research community focus on *Test-time adaptation*. The idea is to adapt the model on the data it is supposed to infer, just before the inference step. This solution enables the network to cope with the distribution or domain shift at test time. However, most of the former methods have only focused on the classification task [6], [7], [8], and mainly on a scenario where distribution shift is modeled in the form of corruptions. Although a more recent method [9] has attempted the synthetic-to-real classification task [10], the performance falls behind that of UDA methods.

In this paper, we propose a depth estimation method that can adapt to the synthetic-to-real domain shift at test time. Similar to former test-time adaptation methods, we adhere to the source-free setup and start from a source domain pretrained network. Given a target domain image for inference, we update our network and adapt to the newly seen domain before the inference step. The update

[1]Korea Advanced Institute of Science and Technology (KAIST), Korea
{djwld93, junmo.kim}@kaist.ac.kr

is done by training our network with a joint combination of three objective functions. The first term is the widely used image gradient smoothness loss, which encourages our depth predictions to be consistent with the image. The second term is the *Ordinal Entropy Reduction* loss, which reduces our uncertainty measure of the current prediction. The last term is the *Logit Order Enforcement* loss, which enforces a proper order of the current ordinal predictions, and aids the network by preventing it from deviating into wrong solutions that result from overly reducing the aforementioned entropy. To validate our method, we compare our method against former unsupervised domain adaptation (UDA) and domain generalization (DG) methods on the KITTI dataset. Our method performs comparable to (DG) or better than (vanilla UDA) former methods while reducing the dependency on training data by $5.8\times$, which is the average value of the maximum scenario ($207.4k$ to $21.6k$) and the minimum scenario ($43.4k$ to $21.6k$). Qualitative and quantitative experiments show the effectiveness of our method. The competent performance and low dependency on data show the real world applicability of our method.

II. Related Work

A. Discretized depth estimation

Depth estimation is a high resolution pixel-wise regression task which therefore, is complex and subtle. In order to relieve this issue, previous methods have proposed to relax the task by dividing the continuous depth values into discrete bins and training it as an ordinal regression formulation [11], [12]. In our work, we have used the method of Fu *et al.* [11] for discretized training, because discretized depth values can be used to form novel concepts that can help in adapting to the unseen data. More recently, Transformer-based architectures have emerged which enabled the network to capture long-range dependencies and use that information to set the bin widths to be input-adaptive [13], [14]. To fairly compare with former domain adaptive depth estimations, and in order to keep test-time adaptation applicable for real-time usage, we do not opt the heavy Transformer-based architectures and implement our method on former CNN-based networks.

B. Unsupervised domain adaptation and Domain generalization in Depth estimation

Unsupervised domain adaptation (UDA) and domain generalization (DG) on the synthetic to realistic scenario have long been investigated in depth estimation. An initial work was that of Kundu *et al.* [15], which opted a residual transfer network and adversarially trained it to be domain invariant. On the other hand, the work of Zheng *et al.* [3], Zhao *et al.* [16] and Lopez-Rodriguez *et al.* [16], [1], all opted an image translation framework. Zheng *et al.* [3] proposed an image translation network and additionally trained the network with source domain depth, target domain smoothness and domain-wise feature consistency. Zhao *et al.* [16] additionally used stereo image pairs and Lopez-Rodriguez *et al.* [1] used a pretrained segmentation backbone to infer pseudo depth based on class and height information. Unlike former work, Chen *et al.* [2] tackled the problem in a DG formulation, by exploiting a variously styled dataset to train the network to become style-invariant. Unlike the aforementioned methods, our work does not depend on excessive additional data, while being comparable to them in terms of performance.

C. Test-time adaptation

The concept of adapting a model at test-time was first attempted by Sun *et al.* [6]. They used the training data to train a network with a self-supervision task [17] and classification. During the test phase, the encoder part of the network was first updated on the test data using the self-supervision branch, and inference was performed thereafter. Wang *et al.* [7] proposed a very concise framework of freezing the entire network and updating the batch normalization [18] parameters only, based on the entropy measured on the test data, which inspired our work. Iwasawa *et al.* [8] proposed to replace the last linear classifier with a prototypical classifier, and the prototypes were obtained by keeping a support set of the target features. In context of prior work, our work is the first work that is targeted on depth estimation which is a dense, pixel-wise regression task.

III. Preliminaries

A. Test-time adaptation setup

In this section we introduce the basic setup of *test-time adaptation*, formally introduced in TENT [7]. The test-time adaptation setup assumes that we only have access to a source data (training data) pretrained model, and access to source data (training data) should not be given (*source-free*), and access to utilize the test data should be granted only once, just before the inference step. During the access, the network may be updated in an *online* mode which performs a single step of update per sample that persists as more samples enter, and an *offline* mode which does a single or multiple steps of update per test sample, performs inference, and resets the update as the next sample arrives. The modes can be seen in *Step 3.* in Fig. 2. In addition to the original setup, we tackle *test-time domain adaptation (TTDA)*, where distribution shift is modeled as domain shift. Throughout this paper, we use both *source data* and *training data* to refer to the synthetic domain data our network is pretrained on, and *target data* and *test data* to refer to the realistic domain data which our network is supposed to infer.

B. Source domain pre-training and DORN

In our work, we train a deep neural network $\boldsymbol{\theta}$ using source domain data $(x_i, y_i), \ldots, (x_{n_S}, y_{n_S})$ where x is an input RGB image, y is a ground truth depth map whose resolution equals x and n_S is the number of source domain data. Given an input image x, the intermediate layers in the network produce multi-scale real-valued depth predictions $\boldsymbol{\theta}(x)_s$ for s in $\mathcal{S}$, where s denotes an element of the set of integer valued downsampling factors $\mathcal{S}$. The last layer outputs $\boldsymbol{\theta}(x)_{dorn}$ and $\boldsymbol{\theta}(x)_1$, which have the same resolution of $x_{(\cdot)}$ and $y_{(\cdot)}$. We

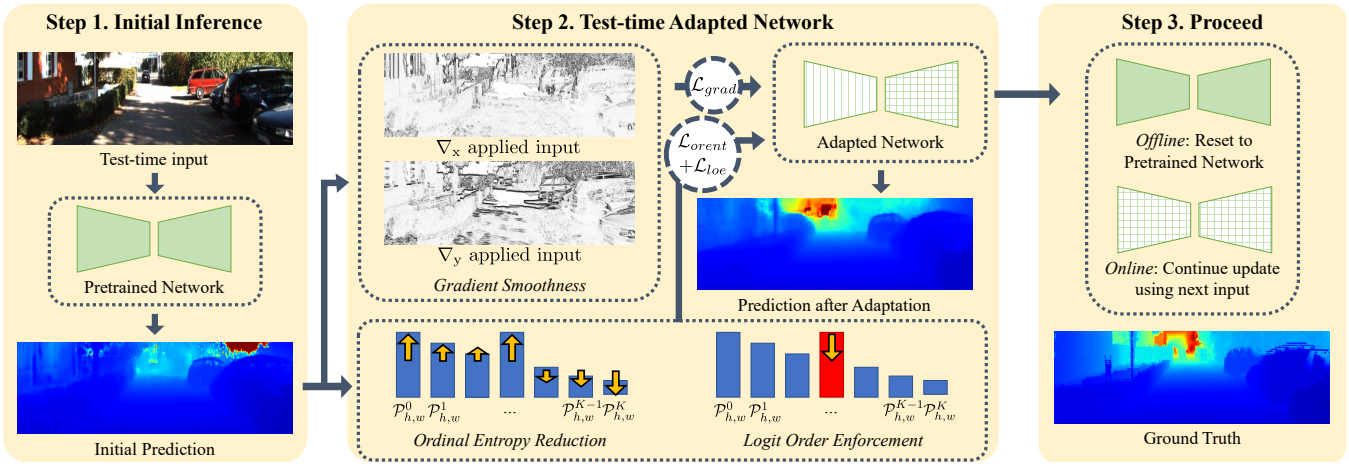

Fig. 2. **Overview of our method. Step 1.** Initial prediction on the newly seen input using a pretrained network. **Step 2.** Adapt the network on the newly seen input using gradient smoothness and our proposed *Ordinal Entropy Reduction* and *Logit Order Enforcement* objective. The adapted network produces improved results. **Step 3.** Proceed to the next input. Reset or keep the network based on whether adaptation is done in *Offline/Online* mode. Ground truth is shown for qualitative comparison.

train the network on the source domain data and obtain the source data pretrained model using

$$\mathcal{L}_{pre} = \sum_{i=1}^{n_s} \mathcal{L}_{reg,ms}(x_i, y_i) + \lambda_{dorn}\mathcal{L}_{dorn}(x_1, y_1) \quad (1)$$

where $\mathcal{L}_{reg,ms}$ denotes the multi-scale regression loss which is defined as the following equation.

$$\mathcal{L}_{reg,ms}(x_i, y_i) = \sum_{s\in\mathcal{S}} \lambda_s \mathbb{E}_{h,w\in I_s} |(\boldsymbol{\theta}(x_i)_s)_{h,w} - (y_{i,s})_{h,w}| \quad (2)$$

λ_s is a balancing hyperparameter across different scales and λ_{dorn} is a balancing hyperparameter for $\mathcal{L}_{dorn}$. h, w are pixel-wise indicators that iterate over the scale-wise pixel set I_s, which result from iterating over the downsampled image or the corresponding downsampled predicted depth $\boldsymbol{\theta}(x_i)_s$. $y_{i,s}$ is the ground truth depth map downsampled by a factor of s. Note that multi-scale outputs are only used for training and not used in any means during the inference or adaptation step, the design of which, was motivated by Eigen *et al.* [19].

dorn is the abbreviation of deep ordinal regression (DORN) from Fu *et al.* [11] and $\mathcal{L}_{dorn}$ trains our network to predict a depth map $\boldsymbol{\theta}(x)_{dorn}$ of size $\mathbb{R}^{K\times h\times w}$ in the form of ordinal regression. The work of DORN discretizes real-valued depth into predefined bins, by setting the i-th depth level threshold t_i for $i \in (0, \ldots, K)$ to be $t_i = \alpha + (\beta - \alpha) * i/K$, but here we set α to 0 and β to the maximum defined depth value for brevity. Then, the network learns to predict a probability distribution for each pixel and threshold, whether the final predicted depth of a given pixel is larger than each of the thresholds. With this formulation, depth estimation is recast into a discrete, bin-wise prediction problem. The paper defines the predicted probability per pixel and threshold as

$$\mathcal{P}^i_{h,w} = P(\hat{l}_{h,w} > i) = \frac{e^{f_{2i+1,h,w}}}{e^{f_{2i+1,h,w}} + e^{f_{2i,h,w}}} \quad (3)$$

which is also the value of $\boldsymbol{\theta}(x)_{dorn}$ at index (i, h, w). The final predicted depth at pixel (h, w) in the DORN formulation is defined as

$$\hat{l}_{h,w} = \sum_{j=0}^{K-1} \mathbb{1}_{\mathcal{P}^j_{h,w} > =0.5} \quad (4)$$

where i is a query depth level and f is a feature map of $\mathbb{R}^{2K\times h\times w}$ used for predicting the probability mentioned above. We refer to the original material if our explanation does not suffice.

IV. Proposed Method

During test-time, our network receives data sampled from a distribution which it has not seen during training. Therefore, the network has to quickly adapt to the new distribution in order to correctly infer depth. Our method is composed of three loss functions that update the network for adaptation. The first loss is a widely used regularization term in unsupervised depth estimation, which enforces the depth predictions to be consistent with the image, in terms of gradients. The second loss refines the current prediction by reducing the uncertainty of the current prediction. The last *Logit Order Enforcement* enforces the current prediction to be ordinally rational, and prevents the network from deviating into wrong solutions that can result from the excessive reduction of the entropy. The overview of our method can be seen in Fig. 2.

A. Gradient smoothness

In this section we first introduce the gradient smoothness loss, which enforces the gradient of depth predictions to be consistent with the gradients of the image. The loss has been widely used in unsupervised [1], [2], [3] and

self-supervised depth estimation [20] frameworks, because it encodes a simple logic that depth changes across boundaries or edges. Updating our network with the loss enables our network to quickly adapt to the newly seen image. The loss is defined as

$$\mathcal{L}_{grad} = \mathbb{E}_{h,w\in I_1} \|\nabla_{\mathrm{x}}(\boldsymbol{\theta}(x)_1)_{h,w}\|_1 e^{-\|\nabla_{\mathrm{x}}(x)_{h,w}\|_1} + \|\nabla_{\mathrm{y}}(\boldsymbol{\theta}(x)_1)_{h,w}\|_1 e^{-\|\nabla_{\mathrm{y}}(x)_{h,w}\|_1} \quad (5)$$

where ∇_{x} and ∇_{y} denote the gradient operator on the x and y direction, respectively. Our indices iterate only over the index set I_1 which is the index set with no downsampling applied. Note that training the prediction $\boldsymbol{\theta}(x)_1$ to be smooth only indirectly affects the discretized prediction result of DORN, by updating the parameters that precede the DORN prediction layer. We have kept the logits of DORN intact because, the following two loss functions are specifically designed for refining the DORN predictions.

B. Ordinal Entropy Reduction

A pioneering work [7] on test-time adaptation has attempted to adapt to the test-time data by reducing the entropy measured from the predictions of the test-time data. Most of the former work in depth estimation have tackled the problem as a regression task which directly regresses the metric depth values, and it is difficult to derive the entropy, or uncertainty from its predictions. However, our method is based on Fu *et al.* [11] as mentioned in Sec. III-B, which allows us to define a pixel-wise uncertainty of the current prediction. We name it *Ordinal Entropy*, denote it as $\mathcal{H}^{ord}_{h,w}$, and its definition and the loss function for reducing it is defined in the next equation.

$$\mathcal{L}_{orent} = \mathbb{E}_{h,w\in I_1} \mathcal{H}^{ord}_{h,w} = \mathbb{E}_{h,w\in I_1} \sum_{i=0}^{K} -\mathcal{P}^i_{h,w} \log(\mathcal{P}^i_{h,w}) \quad (6)$$

Reducing the entropy can refine the current prediction to some extent, by strengthening its current tendency. The direction towards which the ordinal probabilities are updated, can be seen in Fig. 2, denoted by the yellow arrows in the *Ordinal Entropy Reduction* part. But if its current tendency is wrong in the first place, there is no guarantee that it will lead to actual performance improvement. For example, if the current estimate of $\mathcal{P}^i_{h,w}$ is barely larger than 0.5, but its actual value should be 0, entropy optimization will eventually lead in the wrong direction, by pushing its value towards 1. To prevent such catastrophic situations, the hyperparameters of our networks are set to small values such that the network updates are done with care.

C. Logit Order Enforcement

As mentioned before, excessive optimization of *Ordinal Entropy* can lead to problematic situations, by making elementary mistakes that violate the inherent assumptions of the DORN formulation. In this section, we additionally propose an objective function that suppresses such mistakes. From the definition of Eq. 3, we propose that $\mathcal{P}^i_{h,w}$ should monotonically decrease as i increases, because the probability in Eq. 3 for a lower i should encompass the probability for a higher i. However, because of domain shift and entropy optimization, our network can produce mistaken results such as that of the unnaturally high red rectangular-shaped logit of Fig. 2. In order to amend such failure cases, we propose a simple loss function that deters such cases by

$$\mathcal{L}_{loe} = \sum_{j=0}^{K-2} \mathbb{1}_{\mathcal{P}^{j+1}_{h,w} > \mathcal{P}^j_{h,w}} (\mathcal{P}^{j+1}_{h,w} - \mathcal{P}^j_{h,w}) / (\sum_{j=0}^{K-2} \mathbb{1}_{\mathcal{P}^{j+1}_{h,w} > \mathcal{P}^j_{h,w}}). \quad (7)$$

D. Final test-time loss

To sum up, as test-time inputs are passed to a network, our method adapts to the newly seen domain by updating the network with the joint formulation of the aforementioned losses

$$\mathcal{L}_{final} = \lambda_g \mathcal{L}_{grad} + \lambda_o \mathcal{L}_{orent} + \lambda_l \mathcal{L}_{loe} \quad (8)$$

where λ_g, λ_o and λ_l are hyperparameters that control the balance of the losses.

V. Experiments

A. Implementation details

We mainly implement and validate our method on the outdoor synthetic-to-real depth estimation benchmark. The source domain is the VKITTI dataset [21], which features 21k frames of synthetic driving scenes. Our network is first trained on VKITTI and evaluated with adaptation on the KITTI dataset [22], which features images of realistic driving scenes. To show that our method does not overfit on a single adaptation scenario, we validate our method on an indoor synthetic-to-real depth estimation scenario as well. In this scenario, the source domain is the validation set of the SceneNet dataset [23] and the target domain is the NYU Depth V2 [24] dataset. On both of the scenarios, we first start by training our network on the source domain to obtain a source pretrained network. The network consists of an ImageNet [25]-pretrained ResNet50-based [26] encoder and a decoder with transpose convolution, Batch Normalization [18] and ELU [27] activation. The decoder of our network has multiple feature map outlets, for multi-scale and DORN training. The multi-scale set $\mathcal{S}$ is set to $\{1, 2, 4\}$, and two feature outputs from the former part of the decoder participate in multi-scale regression training, except $s = 1$. After the final output of the decoder, two 1×1 convolutions are respectively applied to produce regression predictions for $s = 1$ and DORN.

On the outdoor benchmark, the network is pretrained using the Adam optimizer with batch size, learning rate, weight decay, λ_{dorn}, λ_1, λ_2, λ_4, and K set to 2, 1e-4, 1e-5, 1.0, 1.0, 0.25, 0.25, and 160, respectively. On the indoor benchmark the weight decay, and K are set to 0 and 40, respectively, and the rest of the hyperparameters are the same. β is a dataset dependent value and for VKITTI and SceneNet it is set to 80 and 10, respectively. During training, the images are resized to 192×640 (320×480 for the indoor scenario)

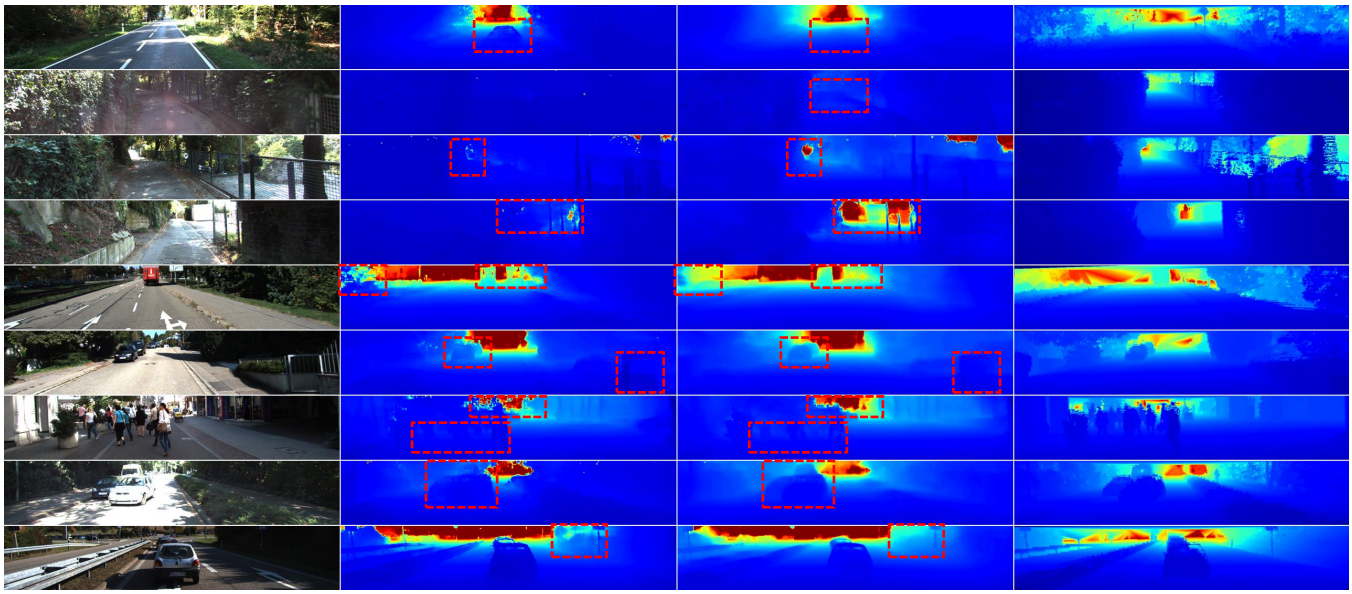

Fig. 3. **Qualitative results on the KITTI test set [19].** Given an input RGB image, our synthetic domain trained network outputs an *Initial Prediction* which suffers from synthetic-to-real domain shift. Our test-time domain adaptation method enhances the initial prediction and helps the network to generalize to an unseen realistic domain, which can be seen in the results in *After TTDA*.

and augmentations such as horizontal flipping, color jittering and ImageNet [25] statistic based normalization are applied. The performance of the pretrained network can be found in the *Src Pretrained* row in Tab. II and Tab. III.

After we obtain a pretrained network, we adapt our network on the test samples and next perform inference. During test-time adaptation on both of the scenarios the learning rate is set to 1e-5 and the encoder and two layers of the decoder are frozen. Regarding the hyperparameters λ_g, λ_o and λ_l that balance the loss functions, we do not tune it and set it to 1.0 as tuning it for performance improvement would be impossible in a real world scenario. Our method is trained on PyTorch [28], using a single RTX2080Ti and half-precision FP16.

B. Quantitative results

Our method is compared against deep learning based UDA and DG methods which perform depth estimation on the KITTI test set [19], without access to ground truth depth data. The methods can be seen in Tab. I. The methods are sorted in descending order, regarding the amount of data used for training. DESC [1] utilizes a segmentation backbone and S2R [2] utilizes a dataset containing various style, denoted by their higher number of 'Data' in Tab. I. AdaDepth [15] and T2Net [3] are evaluated on the vanilla UDA scenario, and thus 43.4k denotes the sum of the number of source and target domain dataset. Because our method only has access to the test set of the target domain only, its number is the lowest. The methods are evaluated using the standard metrics on depth estimation, whose definition can be found in most of the papers for former work, for example, the supplementary material of [2]. Our work has the least dependency on the number of data for generalizing to the new domain, while outperforming vanilla UDA methods and comparably performing to state-of-the-art UDA and DG methods which rely on large-scale additional data for generalization. We conclude that our method design is capable of achieving generalization and reducing dependency on data, at the same time.

C. Qualitative results

The qualitative results on the KITTI test set can be found in Fig. 3, on which we illustrate how our method contributes to test-time domain adaptation. The examples in the first, fifth, sixth and seventh row show that our method is capable of refining wrong and noisy depth predictions that result from the newly seen texture and illumination of the test-time domain. The examples in the second, third and fourth row show that our method has the capability to excavate depth cues in far away regions that were hidden before. The last three rows show that our method can refine object boundaries that were obscure before. Overall, it can be seen that our method is capable of refining initial noisy predictions and recovering fine details, even in the face of domain shift.

D. Outdoor offline and indoor results

On Tab. II we report the performance of our mainly used baseline, and the performance of *Offline* adapted models. The online performance equals that of Tab. I. The trend of our results conforms with that of Sun *et al.* [6] in that the online mode adapted model performs better than the offline mode adapted model. The said work explained this phenomenon by proposing that during the online mode, the model has the flexibility to gradually forget the pretrained knowledge from the training data and adapt to the test-time distribution. On our offline results however, we observe

TABLE I

QUANTITATIVE RESULTS ON THE KITTI TEST SET [19]. OUR METHOD HAS THE LEAST DEPENDENCY ON THE NUMBER OF TRAINING DATA, BUT OUTPERFORMS VANILLA UDA METHODS AND PERFORMS COMPARABLE TO STATE-OF-THE-ART METHODS. *cap* DENOTES THE MAXIMUM VALUE OF DEPTH ON WHICH THE METHODS ARE EVALUATED. THE ↓ / ↑ IN THE SUBSCRIPTS OF THE METRICS DENOTE THAT A *lower/higher* VALUE DENOTES BETTER PERFORMANCE, RESPECTIVELY. *Online* DENOTES THAT OUR NETWORK GETS UPDATED AS IT RECEIVES TEST-TIME DATA INPUT.

Name	Setup	Data	cap	$\text{AbsRel}_{\downarrow}$	$\text{SqRel}_{\downarrow}$	$\text{RMSE}_{\downarrow}$	$\text{RMSE}_{\log\downarrow}$	$\delta_{1\uparrow}$	$\delta_{2\uparrow}$	$\delta_{3\uparrow}$
DESC [1]	UDA + Seg	207.4k	80m	0.162	1.090	5.678	0.244	0.779	0.920	0.968
S2R [2]	DG	100.2k	80m	0.165	1.351	5.695	0.236	0.781	0.931	0.972
AdaDepth [15]	UDA	43.4k	80m	0.214	1.932	7.157	0.295	0.665	0.882	0.950
T2Net [3]	UDA	43.4k	80m	0.171	1.351	5.944	0.247	0.757	0.918	0.969
Ours (Online)	TTDA	**21.6k**	80m	0.168	1.356	5.795	0.239	0.766	0.930	0.974
DESC [1]	UDA + Seg	207.4k	50m	0.155	0.838	4.208	0.227	0.796	0.930	0.973
S2R [2]	DG	100.2k	50m	0.158	1.000	4.321	0.223	0.793	0.939	0.976
AdaDepth [15]	UDA	43.4k	50m	0.203	1.734	6.251	0.284	0.687	0.899	0.958
T2Net [3]	UDA	43.4k	50m	0.165	1.034	4.501	0.235	0.772	0.927	0.972
Ours (Online)	TTDA	**21.6k**	50m	0.160	0.950	4.371	0.225	0.779	0.938	0.978

TABLE II

BASELINE, OFFLINE AND ONLINE PERFORMANCE ON THE KITTI TEST SET [19]. THE ↓ / ↑ IN THE SUBSCRIPTS OF THE METRICS DENOTE THAT A *lower/higher* VALUE DENOTES BETTER PERFORMANCE, RESPECTIVELY. THE SUBSCRIPT OF *Offline* DENOTES THE NUMBER OF UPDATES PER SAMPLE.

Name	$\text{AbsRel}_{\downarrow}$	$\text{SqRel}_{\downarrow}$	$\text{RMSE}_{\downarrow}$	$\text{RMSE}_{\log\downarrow}$	$\delta_{1\uparrow}$	$\delta_{2\uparrow}$	$\delta_{3\uparrow}$
Src Pretrained	0.185	1.802	6.628	0.266	0.745	0.905	0.963
Offline_1	0.180	1.746	6.479	0.260	0.756	0.911	0.965
Offline_2	0.177	1.718	6.377	0.255	0.764	0.914	0.966
Offline_4	0.172	1.678	6.239	0.250	0.773	0.920	0.968
Offline_8	0.168	1.593	6.058	0.243	0.780	0.927	0.971
Offline_{16}	0.171	1.548	5.978	0.242	0.772	0.927	0.972
Online	0.168	1.356	5.795	0.239	0.766	0.930	0.974

TABLE III

QUANTITATIVE RESULTS ON THE NYU DEPTH V2 TEST SET. THE MODEL IS PRETRAINED ON THE VALIDATION SET OF SCENENET. THE DEFINITION OF THE SUBSCRIPTS ARE EQUAL TO THAT OF TAB. II.

Name	$\text{AbsRel}_{\downarrow}$	$\text{SqRel}_{\downarrow}$	$\text{RMSE}_{\downarrow}$	$\text{RMSE}_{\log\downarrow}$	$\delta_{1\uparrow}$	$\delta_{2\uparrow}$	$\delta_{3\uparrow}$
Src Pretrained	0.238	0.248	0.757	0.309	0.626	0.873	0.953
Offline_1	0.235	0.240	0.744	0.303	0.635	0.879	0.954
Offline_2	0.233	0.237	0.738	0.301	0.638	0.880	0.955
Offline_4	0.231	0.234	0.721	0.297	0.643	0.882	0.956
Offline_8	0.231	0.235	0.729	0.291	0.648	0.880	0.956
Offline_{16}	0.235	0.247	0.735	0.286	0.646	0.877	0.955
Online	0.232	0.235	0.737	0.287	0.641	0.875	0.954

that excessively updating the network may lead to *negative* forgetting and thus, lead to performance degradation. The subscripts of the offline models in Tab. II denote the number of update iterations, and it can be seen that until 8 iterations, the performance steadily increases, but beyond that point, the performance degrades, which hints that forgetting has occurred.

To show that our method does not overfit on a certain scenario, we propose to evaluate our method on the indoor scenario, by adapting from SceneNet [23] to NYU Depth V2 [24]. Since there are no papers which were evaluated on this scenario, we only show the baseline and adapted results of our method. Originally, previous methods have proposed to evaluate on an indoor scenario that adapts from SUNCG [29] to NYU Depth V2, but SUNCG became unavailable and therefore, we have attempted this new scenario. The results can be seen in Tab. III. Because the domain gap is larger than the outdoor scenario, the source pretrained model underperforms on the target domain, and therefore, the adaptability is less prominent than that of the outdoor scenario. In addition, the trend is a little bit different from the outdoor scenario; the offline result with 4 and 8 iterations perform slightly better than the online adapted model. We conjecture that the large variation of indoor scenes cause the learned knowledge from a single scene to become less useful as the next scene comes in, thus accelerating negative forgetting and degrading the performance of the online result. Still, it can be seen that our method has a positive effect on adaptation.

VI. CONCLUSION

In this paper, we have proposed a test-time domain adaptation method on the single image depth estimation task. We have proposed to solve this problem because, depth estimation methods do not generalize well to new data and rely on a large amount of training data. Our method overcomes these problems by optimizing various loss functions which effectively adapt the network to an unseen distribution, using a small amount of data. Qualitative and quantitative analyses support the validity of our method.

Acknowledgements This research was supported by the Engineering Research Center Program through the National Research Foundation of Korea (NRF) funded by the Korean Government MSIT (NRF-2018R1A5A1059921) and by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2021-0-02068, Artificial Intelligence Innovation Hub).

REFERENCES

[1] A. Lopez-Rodriguez and K. Mikolajczyk, "Desc: Domain adaptation for depth estimation via semantic consistency," in *British Machine Vision Conference (BMVC)*, 2020.

[2] X. Chen, Y. Wang, X. Chen, and W. Zeng, "S2r-depthnet: Learning a generalizable depth-specific structural representation," 2021.

[3] C. Zheng, T.-J. Cham, and J. Cai, "T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 767–783.

[4] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 3, 2022.

[5] W. Yin, Y. Liu, and C. Shen, "Virtual normal: Enforcing geometric constraints for accurate and robust depth prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[6] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt, "Test-time training with self-supervision for generalization under distribution shifts," in *International conference on machine learning*. PMLR, 2020, pp. 9229–9248.

[7] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, "Tent: Fully test-time adaptation by entropy minimization," in *International Conference on Learning Representations*, 2020.

[8] Y. Iwasawa and Y. Matsuo, "Test-time classifier adjustment module for model-agnostic domain generalization," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 2427–2440.

[9] Y. Liu, P. Kothari, B. van Delft, B. Bellot-Gurlet, T. Mordan, and A. Alahi, "Ttt++: When does self-supervised test-time training fail or thrive?" in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 21 808–21 820.

[10] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko, "Visda: The visual domain adaptation challenge," 2017.

[11] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2002–2011.

[12] R. Diaz and A. Marathe, "Soft labels for ordinal regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4738–4747.

[13] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4009–4018.

[14] Z. Li, X. Wang, X. Liu, and J. Jiang, "Binsformer: Revisiting adaptive bins for monocular depth estimation," *arXiv preprint arXiv:2204.00987*, 2022.

[15] J. N. Kundu, P. K. Uppala, A. Pahuja, and R. V. Babu, "Adadepth: Unsupervised content congruent adaptation for depth estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2656–2665.

[16] S. Zhao, H. Fu, M. Gong, and D. Tao, "Geometry-aware symmetric domain adaptation for monocular depth estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9788–9798.

[17] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations*, 2018.

[18] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, 2015, pp. 448–456.

[19] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[20] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3828–3838.

[21] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *CVPR*, 2016.

[22] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[23] J. McCormac, A. Handa, S. Leutenegger, and A. J.Davison, "Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation?" 2017.

[24] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[29] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017.