Deep Domain Adversarial Adaptation for Photon-Efficient Imaging

Yiwei Chen,[1,2] Gongxin Yao,[1,2] Yong Liu,[1,2] Hongye Su,[1,2] Xiaomin Hu,[3,4] and Yu Pan[1,2,*]

[1] *State Key Laboratory of Industrial Control Technology, Hangzhou 310027, People's Republic of China*
[2] *College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, People's Republic of China*
[3] *CAS Key Laboratory of Quantum Information, University of Science and Technology of China, Hefei 230026, People's Republic of China*
[4] *CAS Center For Excellence in Quantum Information and Quantum Physics, University of Science and Technology of China, Hefei 230026, People's Republic of China*

(Received 14 May 2022; revised 1 August 2022; accepted 11 October 2022; published 16 November 2022)

Photon-efficient imaging with the single-photon light detection and ranging captures the three-dimensional structure of a scene by only a few detected signal photons per pixel. However, the existing computational methods for photon-efficient imaging are pretuned on a restricted scenario or trained on simulated datasets. When applied to realistic scenarios whose signal-to-background ratios and other hardware-specific properties differ from those of the original task, the model performance often significantly deteriorates. In this paper, we present a domain adversarial adaptation design to alleviate this domain shift problem by exploiting unlabeled real-world data, with significant resource savings. This method demonstrates superior performance on simulated and real-world experiments using our home-built up-conversion single-photon imaging system, which provides an efficient approach to bypass the lack of ground-truth depth information in implementing computational imaging algorithms for realistic applications.

DOI: 10.1103/PhysRevApplied.18.054048

I. INTRODUCTION

Single-photon light detection and ranging (LiDAR) systems have achieved dramatic success in three-dimensional (3D) imaging over the past decades [1–5]. Benefiting from the high sensitivity of single-photon avalanche diode (SPAD) detector [6,7], single-photon LiDAR provides better precision over a longer distance as compared to the traditional systems based on the photomultiplier tube [8]. In practice, a photon counting histogram is collected at each scanned point with the time-correlated single-photon counting (TCSPC) technique [9], which records the timestamps of emitted and reflected photons for the time-of-flight (TOF) calculation. However, due to practical constraints on optical flux and integration time, the average number of detected photons per pixel, including signal and noise counts, could be less than a few dozens. This calls for photon-efficient imaging [10,11], whose goal is to recover a precise depth image from a sparsely distributed histogram of few photon counts.

Machine-learning methods, including statistical [11,12] and deep-learning models [13–15], have been proposed for photon-efficient imaging. Particularly, deep-learning models have demonstrated remarkable performance in terms of reconstruction accuracy and inference time. Deep-learning models need huge amounts of data for training, whereas it is extremely difficult to acquire enough annotated real-world data. As a result, the existing models are trained on simulated datasets for which the signal-to-background ratios (SBRs) and other hardware-specific properties have to be assumed. For example, SBR could be affected by light condition, object distance, or noise inherent in the hardware, such as the dark counts of SPAD and pump light [16]. Hardware-specific properties also include the shape of a pulse [17]. Therefore, the model trained on simulated dataset may not generalize well to realistic hardware or environment for which SBR or pulse shape are different, causing the domain shift problem [18–20]. Although a calibration step can be conducted to estimate the realistic parameters, regenerating enough simulated data and retraining the model to fit the target domain requires tens of hours and a huge amount of computational resources, which is highly expensive and unscalable. Considering that ground-truth label is basically inaccessible in a real-world experiment, training the deep-learning model with real-world data is even more impossible.

Deep domain adversarial adaptation [21,22] is a promising technique for deep-learning models to transfer general knowledge across similar machine-learning tasks. Domain-adversarial neural network (DANN) [23] is

[*] ypan@zju.edu.cn

proposed to learn a feature extractor to extract domain-independent features in an adversarial regime. This approach and its variants have been applied to many research fields, such as image classification [23,24] and semantic segmentation [25].

In this paper, we propose a deep domain adversarial adaptation method with an effective network structure to tackle the domain shift problem in photon-efficient imaging. This method requires only dozens of extra training iterations with a small amount of unlabeled data from the target domain for the model to adapt to different SBRs or hardware-specific properties. Extensive experiments on the simulated datasets demonstrate the efficiency of our method for the model to adapt to different SBRs. Moreover, the real-world experiment based on our home-built single-photon imaging system shows that our method improves the baseline model performance by a remarkable margin, while other statistical and deep-learning algorithms do not generalize well to the realistic scenario.

II. METHOD

The domain adversarial adaptation method is adopted mainly from [23] and its schematic diagram is shown in Fig. 1. The feature extractor, denoted by $\mathcal{F}$, learns the transformation that maps both domains into a common feature space. The purpose of this transformation is to extract domain-independent features from the inputs to confuse the discriminator, while the discriminator, denoted by $\mathcal{D}$, tries to determine whether these features are from the source or the target domain. This process, as marked by the dashed box, corresponds to a minimax two-player game, which forces $\mathcal{F}$ to focus on capturing the similarity of the two domains. Meanwhile, the reconstructor denoted by $\mathcal{R}$ learns to estimate the ground-truth depth from the source features.

In principle, this scheme can be applied to any deep-learning model as long as a suitable feature space has been defined. Nevertheless, existing deep-learning models either involve complex attention operations [14] or residual connections [15], which makes it difficult to find an encoder-decoder division for adversarial adaptation training. Therefore, we propose a simple yet efficient

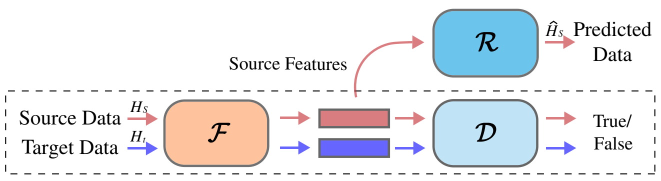

FIG. 1. Schematic diagram of the deep domain adversarial adaptation scheme. $\mathcal{F}$, feature extractor; $\mathcal{D}$, discriminator; $\mathcal{R}$, reconstructor. We use red and blue colors to indicate the pipelines on the source and target domain, respectively.

encoder-decoder network structure design for the domain adaptation task. Firstly, inspired by Refs. [26] and [27], we design a spatiotemporal block (ST block), which consists of four branches based on the spatiotemporal convolution (see Appendix A 1 for more details). $\mathcal{F}$ is built by stacking eight modules, each composed of a ST block and a 3D pooling layer. Then $\mathcal{R}$ comprises seven deconvolutional layers and a $1 \times 1 \times 1$ convolutional layer. In particular, each convolutional and deconvolutional layer is followed by a group normalization layer and a rectified linear unit (ReLU) activation. We name the whole composition of $\mathcal{F}$ and $\mathcal{R}$ as the spatiotemporal inception network (STIN). The structure of STIN is given in Appendix A 2.

A. Pretraining

STIN is first trained on the source domain as a baseline model. Here the depth estimation is taken as a classification problem, in which STIN aims at correctly classifying each pixel into a depth category that matches its ground-truth depth [28]. The photon-counting histograms $H \in \mathbb{R}^{T\times N_x\times N_y}$ for the source domain are generated by simulation from the ground-truth depth images $Z \in \mathbb{R}^{N_x\times N_y}$, where T, N_x, and N_y are the number of time bins, the horizontal and vertical resolutions, respectively. The prediction of the model is denoted as $\hat{H} = \mathcal{R}(\mathcal{F}(H)) \in \mathbb{R}^{T\times N_x\times N_y}$. Note that Z has to be further processed by the following rounding function for classification:

$$z_{i,j} = [z_{i,j}] = \max\left\{n \in \mathbb{Z}_0^+ \mid n \le \frac{2z_{i,j}}{\Delta \times c}\right\}, \tag{1}$$

where c is the light speed and Δ is the size of time bin. The cross-entropy (CE) loss between $\hat{H}$ and Z is given by

$$\mathcal{L}_{\mathrm{CE}}(\hat{H}, Z) = \sum_{i,j} \mathcal{L}_{\mathrm{CE}}(\hat{\mathbf{h}}_{\mathbf{i,j}}, \mathbf{z}_{\mathbf{i,j}}^{(\mathrm{o})}), \tag{2}$$

which is adopted from Ref. [28], and $\mathbf{z}_{\mathbf{i,j}}^{(\mathrm{o})} \in \mathbb{R}^{\mathbf{T}}$ is the one-hot encoding of $z_{i,j}$. Another total variation (TV) regularization term $\mathcal{L}_{\mathrm{TV}}$ for smoothing the prediction [15] is added to make the total loss function as

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{TV}} \times \mathcal{L}_{\mathrm{TV}}, \tag{3}$$

where λ_{TV} is the weight of the TV term. The details of $\mathcal{L}_{\mathrm{CE}}$ and $\mathcal{L}_{\mathrm{TV}}$ are given in Appendix A 3. $\mathcal{F}$ and $\mathcal{R}$ are trained by standard backpropagation with respect to the loss Eq. (3) on the source domain.

B. Domain adversarial training

Domain adversarial training is achieved by alternatively minimizing the following loss functions:

$$\mathcal{L}(H_s, Z_s) = \mathcal{L}(\mathcal{R}(\mathcal{F}(H_s)), Z_s), \tag{4}$$

$$\mathcal{L}_{\text{adv}}(H_s, H_t) = \log \mathcal{D}(\mathcal{F}(H_s)) - \log(1 - \mathcal{D}(\mathcal{F}(H_t)), \tag{5}$$

where H_s and Z_s are the sample and ground-truth depth from the source domain, and H_t is the sample from the target domain. It should be noted that the ground-truth depth of the target domain is not needed for domain adversarial training. In each training iteration, $\mathcal{D}$ is firstly optimized by minimizing the domain adversarial loss defined in Eq. (5) to extract domain-invariant features. Then $\mathcal{F}$ and $\mathcal{R}$ are simultaneously optimized by minimizing the total loss function given by

$$\mathcal{L}_{\text{total}} = \mathcal{L} - \lambda_a \times \mathcal{L}_{\text{adv}} \tag{6}$$

to improve the reconstruction performance, with λ_a being the weight of adaptation. The detailed implementation of DANN training is shown in Algorithm 1, in which we denote S and T as the training sets on the source and target domain, respectively.

The goal of the discriminator is to distinguish between the source and target domain from the latent features. Thus, Z has to maintain the temporal dimension of the features, while the spatial dimension of the features needs to be blurred. To do this, the latent features first go through a 3D pooling layer with a kernel size of $1 \times 8 \times 8$. After that, three fully connected layers will map these features to a scalar that classifies the input into either the source or the target domain.

After the adversarial training, the target depth image denoted by $\hat{Z}_t$ is predicted by the Softargmax function [13] as

$$\begin{aligned}[\hat{z}_t]_{i,j} &= \frac{\Delta \times c}{2} \text{Softargmax}([\hat{\mathbf{h}}_{\mathbf{t}}]_{\mathbf{i,j}}) \\ &= \frac{\Delta \times c}{2} \sum_{k=1}^{T} k \times [\hat{h}_t]_{k,i,j},\end{aligned} \tag{7}$$

where $\hat{H}_t = \mathcal{R}(\mathcal{F}(H_t))$. Note that the Softargmax function is equivalent to maximum-likelihood estimation (MLE) [29].

```
Input:
 1: samples: $\mathbf{S} = \{(H_s{}^{(i)}, Z_s{}^{(i)})\}_{i=1}^{N}$ and
 2:          $\mathbf{T} = \{H_t{}^{(i)}\}_{j=1}^{N'}$;
 3: adaptation weight $\lambda_a$;
 4: pre-trained $\mathcal{F}$ and $\mathcal{R}$ on the source domain;
 5: randomly initialized $\mathcal{D}$;
Output: new $\mathcal{F}$ and $\mathcal{R}$;
 6: while stopping criterion is not met do
 7:     for $i \leftarrow 1, \cdots, N$ do
 8:         Randomly sample $j$ from $\{1, \cdots, N'\}$;
 9:         Calculate $\mathcal{L}_{\text{adv}}(H_s^{(i)}, H_t^{(j)})$;
10:         Update $\mathcal{D}$ by minimizing $\mathcal{L}_{\text{adv}}$;
11:         Calculate the total loss by
12:             $\mathcal{L}_{\text{total}}(H_s^{(i)}, Z_s^{(i)}, H_t^{(j)}) \leftarrow$
                $\mathcal{L}(H_s^{(i)}, Z_s^{(i)}) - \lambda_a \cdot \mathcal{L}_{\text{adv}}(H_s^{(i)}, H_t^{(j)})$;
13:         Update $\mathcal{F}$ and $\mathcal{R}$ by minimizing $\mathcal{L}_{\text{total}}$;
14:     end for
15: end while
```

Algorithm 1. Implementation of DANN

III. RESULTS

A. Metrics and implementation details

The following metrics are used for the quantitative evaluation.

(a) Root-mean-squared error (RMSE), defined as

$$E_{\text{RMS}}(Z, \hat{Z}) = \sqrt{\frac{1}{N_x N_y} \sum_{i}^{N_x} \sum_{j}^{N_y} \left(z_{i,j} - \hat{z}_{i,j}\right)^2}. \tag{8}$$

(b) Absolute relative difference (ARD), defined as

$$D_{\text{AR}}(Z, \hat{Z}) = \frac{1}{N_x N_y} \sum_{i}^{N_x} \sum_{j}^{N_y} \frac{\left|z_{i,j} - \hat{z}_{i,j}\right|}{z_{i,j}}. \tag{9}$$

All the models are implemented using the deep-learning library Pytorch [30] and trained on a single GTX 1080ti GPU. We adopt the Adam optimizer [31] with a base learning rate of 0.001. λ_{TV} and λ_a are set to be 0.001 and 0.1, respectively. The code for implementing our model is available at Ref. [32].

B. Simulated experiments

The validation experiments are firstly conducted on the simulated dataset. The goal is to adapt the model pretrained on low-noise domain to high-noise domain. 15 940 low-noise samples with ground-truth depth information and 42 high-noise samples without ground-truth information are generated from NYU v2 dataset [33] as the training set using the observation model described in Appendix C. The total number of time bins is 1024 and the bin size is 80 ps. The waveform of the emitted pulse is assumed to have an ideal Gaussian shape with the full width at half maximum of 400 ps. Constrained by GPU memory size, we divide the training samples into patches of size [1024, 32, 32] and set the batch size to 6. The SBRs for generating the low-noise and high-noise samples are {2:2, 5:2, 10:2} and {2:50, 2:100}, respectively. Here a SBR of n:m stands for the ratio of the average number of signal photons to noise photons per pixel. The test set is

TABLE I. Comparisons of the baseline and DANN-based model on simulated datasets. Best results are highlighted in bold. The percentage difference from the baseline is also shown in the parentheses.

	Baseline		DANN-based STIN	
SBR	RMSE	ARD	RMSE	ARD
2:2	**0.026**	**0.018**	0.031(↑ 19.6%)	0.019(↑ 3.8%)
2:50	0.121	0.036	**0.076(↓ 37.1%)**	**0.028(↓ 24.9%)**
2:100	0.388	0.147	**0.088(↓ 77.3%)**	**0.040(↓ 72.5%)**

composed of samples with SBRs of {2:2, 2:50, 2:100}, which are generated from Middlebury dataset [34] with a resolution of 512×512. As noted before, the STIN is first trained on the low-noise dataset as the baseline model. The training takes about 10 h. Then the trained model is adapted to the high-noise domain using the 42 high-noise samples by domain adversarial training, which takes only about 10 min. In testing, the high-resolution input image is divided into small patches for depth prediction and then the outputs are concatenated into a large image.

As shown in Table I, the baseline model does not generalize well to the high-noise test samples with SBR of 2:50 or 2:100, whereas the DANN-based adaptation method achieves a significant improvement at low SBRs. Particularly, the DANN-based model outperforms the baseline by 77.3% in RMSE and 72.5% in ARD when SBR is 2:100. At the high SBR of 2:2, the performance of the DANN-based model is worse than the baseline model, since the baseline is trained solely with this SBR. Meanwhile, the DANN-based model has to adapt to high-noise samples, which inevitably decreases its performance at SBR = 2:2 since the model needs to balance between two domains.

Figure 2 shows the predicted depth images for the Laundry scene from the test set. We see that the baseline can precisely recover the depth image at high SBRs, whereas many erroneous regions appear when SBR decreases. In contrast, the performance of the DANN-based model is much more robust to the decrease in SBR.

We conduct extra experiments to study the sensitivity of λ_{TV} and λ_a; see Table II. When the weight of adaptation is too small, such as $\lambda_a = 0.01$, the improvement in performance is not significant. Otherwise, the protocol is not sensitive to the regularization coefficient λ_{TV}.

C. Real-world experiments

Next, we conduct real-world experiment by building a single-photon imaging system, whose schematic diagram is shown in Fig. 3. The system is based on an up-conversion process that transforms the near-infrared signal photon to the visible regime for detection [16]. The picosecond laser source emits synchronized pump pulses (1565.5 nm) and signal pulses (1554.1 nm) with the repetition rate of 600 MHz. The signal pulses are propagated to the programmable gimbal-less two-axis microelectromechanical system (MEMS) mirror through an optical transceiver for scanning. In the meantime, the pump pulses are delayed in the programmable motorized ODL. By adjusting the delay time such that the signal and pump pulses coincide (the optical delay time matches the target distance), the up-conversion light of 779.8 nm for photon detection is generated by the sum-frequency generation process in the PPLN waveguide. Then the generated light is passed through several home-built FBFs to filter out the noise, before finally being detected by the SPAD detector.

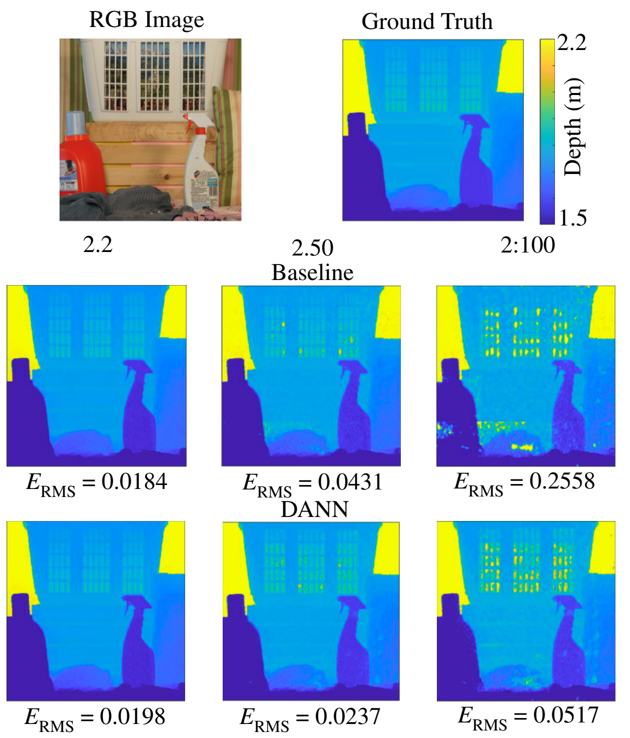

FIG. 2. The results of a random selected scene named Laundry with different SBRs in the Middlebury dataset. The first row includes the RGB image and ground truth. The second and third rows include the depth images predicted by STIN and DANN-based STIN, respectively.

TABLE II. RMSEs for different λ_{TV} and λ_a on the simulated dataset with the SBR of 2:50. The best result is highlighted in bold.

	$\lambda_{\mathrm{TV}} = 0.01$	$\lambda_{\mathrm{TV}} = 0.001$	$\lambda_{\mathrm{TV}} = 0.0001$
$\lambda_a = 1$	0.078	0.077	0.085
$\lambda_a = 0.1$	0.078	**0.076**	0.082
$\lambda_a = 0.01$	0.113	0.119	0.114

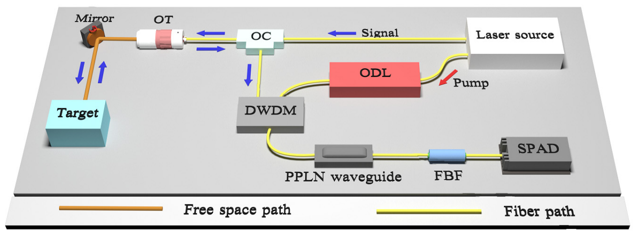

FIG. 3. Schematic diagram of our imaging system. ODL, optical delay line; OC, optical circulator; OT, optical transceiver; DWDM, dense wavelength division multiplexing; PPLN waveguide, periodically poled lithium niobate waveguide; FBF, fiber bandpass filter; SPAD, single-photon avalanche diode.

We use a field-programmable gate array (FPGA) to count the number of detected photons at each time-delay step. A desktop computer is employed to control the MEMS mirror and ODL. Unlike Ref. [16], we use commercially packaged PPLN waveguide instead of bulk PPLN crystal in the conversion module. Compared with bulk crystal, the waveguide has higher up-conversion efficiency with a more stable all-fiber structure.

The detected noise photons as shown in Fig. 4(a) come from the environment, dark counts of the SPAD and the pump light. In particular, the frequency-doubled photons of the pump light that cannot be completely eliminated by the optical filters constitute the major source of noise, whereas the simulation only assumes a uniform distribution for ambient noise. Besides, it can be observed that there is a visible difference between the real-world waveform of detected photons in Fig. 4(b) and the simulated waveform in Fig. 4(c). Therefore, the noise source, statistics and detected waveform of our system substantially differ from the ideal simulation, which makes it a perfect candidate for the validation of the domain adaptation method.

The 2022 Asian Games mascot Chenchen is placed outdoors and about 5 meters away from the optical transceiver. The ODL is scanned in 1 ps steps over a range of 1024 ps. In order to simulate photon-efficient imaging, we set the acquisition time for each time delay step as 1 μs such that no more than dozens of signal and noise photons are recorded at each pixel. The baseline STIN is pretrained on the simulated dataset with SBRs of {2:10, 2:50, 2:100}. By adjusting the number of optical filters, the overall SBR of our system is measured to be 0.103, which does not overlap with the SBRs for pretraining. The pretrained STIN is then tuned by several patches with the resolution of $[1024, 32, 32]$ from the real-world data using the proposed domain adversarial adaptation method. The domain adaptation process takes only several minutes, without using any ground-truth depth information in the real world. In evaluation, by increasing the acquisition time at each time-delay step to 1 ms like Ref. [14], we collect enough signal photon counts for the precise depth estimation of each pixel that approximates the ground truth using MLE. As shown in Fig. 5, our method achieves a significant visual improvement when

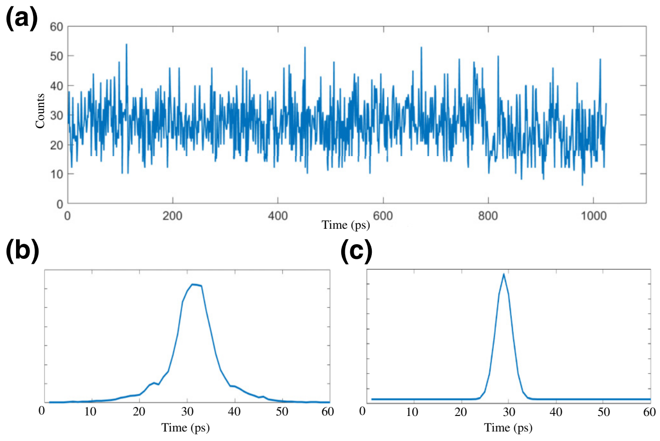

FIG. 4. (a) Noise photon counts of our system. (b) The normalized real-world pulse shape of our system. (c) The normalized ideal pulse shape used for simulation.

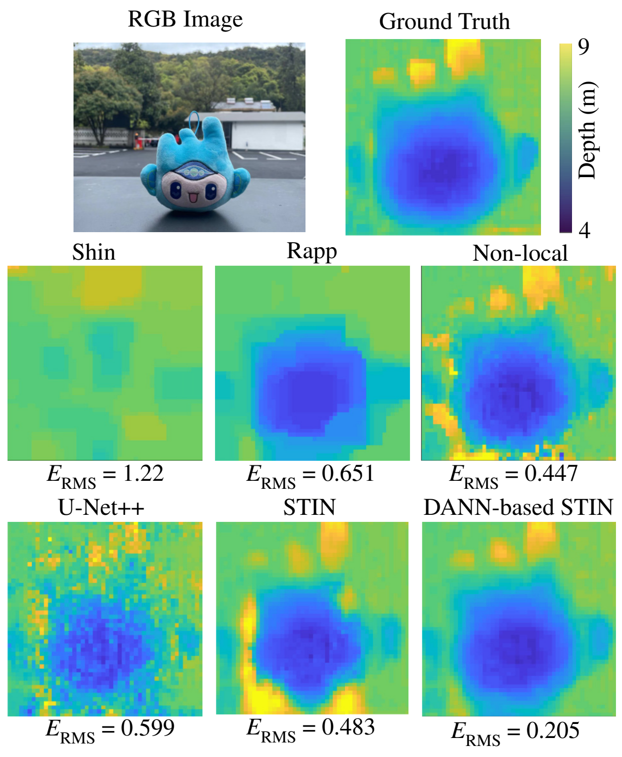
RGB Image
Ground Truth
Depth (m)
9
4
Shin
Rapp
Non-local
$E_{\mathrm{RMS}} = 1.22$
$E_{\mathrm{RMS}} = 0.651$
$E_{\mathrm{RMS}} = 0.447$
U-Net++
STIN
DANN-based STIN
$E_{\mathrm{RMS}} = 0.599$
$E_{\mathrm{RMS}} = 0.483$
$E_{\mathrm{RMS}} = 0.205$

FIG. 5. The reconstruction results of 2022 Asian Games mascot Chenchen with a spatial resolution of 48 × 48. The estimated ground-truth data are only used for evaluation but not for adaptation.

TABLE III. Comparison with respect to RMSE, ARD, and accuracy with δ (defined in Appendix B) of the deep-learning methods on the real-world experiment. Best results are highlighted in bold.

	Error		Accuracy with δ	
Models	RMSE (cm)	ARD	$\delta = 1.01$	$\delta = (1.01)^2$
Nonlocal	0.447	0.0015	33.3%	53.9%
U-Net++	0.599	0.0021	19.5%	30.3%
STIN	0.483	0.0012	30.5%	61.3%
DANN	**0.205**	**0.0008**	**40.8%**	**66.3%**

compared to other state-of-the-art competitors, including two statistical methods [11,12], two deep-learning models [14,15], and the baseline model. These competitors all have shown good performance on the simulated dataset as detailed in Appendix B and poor generalizability in real experiment, which highlights the necessity of domain adaptation. We provide a more comprehensive comparison in Table III. In particular, the DANN-based STIN achieves the highest imaging quality with a decrease in RMSE of 57.5%, 65.8% and 54.1% and in ARD of 33.3%, 61.9% and 46.6%, as compared to baseline, U-Net++, and nonlocal net, respectively. In addition, DANN achieves significant improvements in the metric accuracy with δ.

IV. CONCLUSION

We propose an efficient domain adversarial adaptation method to improve the depth estimation precision for a 3D scene, without using any ground-truth depth information of the target domain. Experimental results on simulated dataset and our home-built imaging system have verified the effectiveness of our method. It has been shown that although the pretrained models may be effective on the simulated dataset, their performance may deteriorate significantly if domain adaptation is not conducted. Considering that obtaining sufficient data, either real-world data with ground-truth labels or simulated data, to retrain the model in the target domain is extremely time consuming or merely impossible, the proposed approach of this paper is able to provide significant resource savings for the practical deployment of deep-learning models in photon-efficient imaging applications.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (Grant No. 62173296).

APPENDIX A: DETAILS OF STIN

1. ST block

The first branch consists of a $1 \times 1 \times 1$ 3D convolutional layer. The second branch is composed of a $1 \times 1 \times 1$ convolutional layer followed by an $n_t \times 1 \times 1$ and a $1 \times n_s \times n_s$ 3D convolutional layer. Here n_t and n_s are the kernel size for the temporal and spatial dimensions, respectively. The third branch swaps the locations of the temporal and spatial convolutions in the second branch. The fourth branch is composed of a $1 \times 1 \times 1$ convolutional layer, two $1 \times n_s \times n_s$ convolutional layers and an $n_t \times 1 \times 1$ convolutional layer. Finally, the outputs of four branches are concatenated together along the channel dimension. For example, if the size of the input is $[B, C_i, T, H, W]$, then the size of the final output is $[B, 4 \times C_b, T, H, W]$, where C_b is the number of output channels on each branch.

TABLE IV. Structure of the feature extractor.

Layer	Kernel size	Output size
Input	$\cdots$	$[1, 1024, 32, 32]$
ST block	$7 \times 3 \times 3$	$[4, 1024, 32, 32]$
Max pooling	$2 \times 1 \times 1$	$[4, 512, 32, 32]$
ST block	$7 \times 3 \times 3$	$[8, 512, 32, 32]$
Max pooling	$2 \times 1 \times 1$	$[8, 256, 32, 32]$
ST block	$7 \times 3 \times 3$	$[12, 256, 32, 32]$
Max pooling	$2 \times 1 \times 1$	$[12, 128, 32, 32]$
ST block	$7 \times 3 \times 3$	$[16, 128, 32, 32]$
Max pooling	$2 \times 1 \times 1$	$[16, 64, 32, 32]$
ST block	$7 \times 3 \times 3$	$[24, 64, 32, 32]$
Max pooling	$2 \times 1 \times 1$	$[24, 32, 32, 32]$
ST block	$7 \times 3 \times 3$	$[32, 32, 32, 32]$
Max pooling	$2 \times 1 \times 1$	$[32, 16, 32, 32]$
ST block	$7 \times 3 \times 3$	$[40, 16, 32, 32]$
Max pooling	$2 \times 1 \times 1$	$[40, 8, 32, 32]$
ST block	$7 \times 3 \times 3$	$[48, 8, 32, 32]$

2. Network structure

We summarize the detailed structure of the feature extractor, the discriminator and the reconstructor in Tables IV, V, and VI, respectively.

TABLE V. Structure of the discriminator. FC denotes the fully connected layer.

Layer	Kernel size	Output size
Input	$\cdots$	$[48, 8, 32, 32]$
Average pooling	$1 \times 8 \times 8$	$[48, 8, 4, 4]$
Reshape	$\cdots$	$[6144]$
FC	$\cdots$	$[512]$
FC	$\cdots$	$[128]$
FC	$\cdots$	$[1]$

TABLE VI. Structure of the reconstructor.

Layer	Kernel size	Output size
Input	...	[48, 8, 32, 32]
3D Deconv	6 × 3 × 3	[40, 16, 32, 32]
3D Deconv	6 × 3 × 3	[32, 32, 32, 32]
3D Deconv	6 × 3 × 3	[24, 64, 32, 32]
3D Deconv	6 × 3 × 3	[16, 128, 32, 32]
3D Deconv	6 × 3 × 3	[12, 256, 32, 32]
3D Deconv	6 × 3 × 3	[8, 512, 32, 32]
3D Deconv	6 × 3 × 3	[4, 1024, 32, 32]
3D Conv	1 × 1 × 1	[1, 1024, 32, 32]

3. The loss functions

The CE loss for $\boldsymbol{a} \in \mathbb{R}^T$ and $\boldsymbol{b} \in \mathbb{R}^T$ is defined as

$$\mathcal{L}_{\text{CE}}(\boldsymbol{a}, \boldsymbol{b}) = -\sum_{t}^{T} a_t \log b_t. \tag{A1}$$

The TV loss for the input $Z \in \mathbb{R}^{N_x \times N_y}$ is defined as

$$\mathcal{L}_{\text{TV}}(\hat{Z}) = \sum_{i,j}(|\hat{z}_{i+1,j} - \hat{z}_{i,j}| + |\hat{z}_{i,j+1} - \hat{z}_{i,j}|). \tag{A2}$$

APPENDIX B: FURTHER RESULTS ON SIMULATED DATASET

The comparison of Shin [11], Rapp [12], nonlocal net [14], U-net++ [15], and STIN (ours) are shown in Table VII. An additional metric named the accuracy at a given threshold δ is defined as

$$\text{percentage of } \hat{D},$$

$$\text{such that } \frac{1}{N_x N_y}\sum_{i}^{N_x}\sum_{j}^{N_y} \max\left(\frac{d_{i,j}}{\hat{d}_{i,j}}, \frac{\hat{d}_{i,j}}{d_{i,j}}\right) < \delta. \tag{B1}$$

APPENDIX C: OBSERVATION MODEL

The detected photon counts are modeled as a linear mixture of signal and background photons corrupted by Poisson noise [11]. Given the laser pulse waveform $s(t)$, the Poisson-process rate function for the (i,j)th pixel of the object to be detected is

$$r_{i,j}(t) = \eta a_{i,j} s\left(t - \frac{2z_{i,j}}{c}\right) + n_{i,j}, \tag{C1}$$

where $n_{i,j}$ and $\eta \in (0, 1]$ are the average count of the noise and the detection efficiency of the SPAD detector, respectively. Here, $a_{i,j}$ and $z_{i,j}$ are the albedo and the ground-truth depth of the (i,j)th scanned point, respectively. Assume that the size of time bin is Δ, the total number of time bins is T and the number of illuminations is n_t, the statistical distribution for each time bin is formulated as

$$h_{i,j,t} \sim \text{Possion}(n_t \int_{(t-1)\Delta}^{t\Delta} r_{i,j}(x)dx), \ t = 1, \dots, T. \tag{C2}$$

The histograms of pixels are concatenated together to form a photon counting cube of a size $[T, N_x, N_y]$, where T, N_x, N_y are the number of time bins, the horizontal and vertical resolutions, respectively.

TABLE VII. Comparison results on the simulated dataset. Best results are highlighted in bold.

	SBR = 2:10				
	Error (lower is better)		Accuracy with δ (higher is better)		
Models	RMSE (m)	ARD	$\delta = 1.01$	$\delta = (1.01)^2$	$\delta = (1.01)^3$
Shin	3.2225	3.1995	0	0	0
Rapp	0.0610	0.0350	43.45%	71.05%	82.74%
Nonlocal Net	0.0244	0.0106	55.30%	93.68%	97.58%
U-Net++	0.0278	**0.0087**	**78.79%**	93.46%	96.22%
STIN	**0.0211**	0.0094	62.15%	**94.76%**	**99.35%**
	SBR = 2:50				
Shin	4.1783	4.1723	0	0	0
Rapp	0.0676	0.0399	39.87%	67.71%	79.42%
Nonlocal Net	0.0297	0.0113	55.58%	91.66%	96.16%
U-Net++	0.0787	0.0174	**70.96%**	85.11%	88.84%
STIN	**0.0249**	**0.0102**	60.01%	**92.43%**	**97.08%**
	SBR = 2:100				
Shin	4.2955	4.2901	0	0	0
Rapp	0.0726	0.0442	38.08%	64.83%	76.61%
Nonlocal Net	0.0329	0.0122	52.66%	89.12%	95.05%
U-Net++	0.0795	0.0172	**70.57%**	85.36%	89.16%
STIN	**0.0305**	**0.0112**	57.92%	**90.38%**	**95.66%**

[1] A. Kirmani, D. Venkatraman, D. Shin, A. Colaço, F. N. Wong, J. H. Shapiro, and V. K. Goyal, First-photon imaging, Science **343**, 58 (2014).

[2] R. Lussana, F. Villa, A. Dalla Mora, D. Contini, A. Tosi, and F. Zappa, Enhanced single-photon time-of-flight 3D ranging, Opt. Express **23**, 24962 (2015).

[3] Y. Altmann, S. McLaughlin, M. J. Padgett, V. K. Goyal, A. O. Hero, and D. Faccio, Quantum-inspired computational imaging, Science **361**, eaat2298 (2018).

[4] G. Musarra, A. Lyons, E. Conca, Y. Altmann, F. Villa, F. Zappa, M. Padgett, and D. Faccio, Non-Line-of-Sight Three-Dimensional Imaging with a Single-Pixel Camera, Phys. Rev. Appl. **12**, 011002 (2019).

[5] Z.-P. Li, X. Huang, Y. Cao, B. Wang, Y.-H. Li, W. Jin, C. Yu, J. Zhang, Q. Zhang, C.-Z. Peng, Feihu Xu, and Jian-Wei Pan, Single-photon computational 3D imaging at 45 km, Photonics Res. **8**, 1532 (2020).

[6] R. H. Hadfield, Single-photon detectors for optical quantum information applications, Nat. Photonics **3**, 696 (2009).

[7] K. L. Nicolich, C. Cahall, N. T. Islam, G. P. Lafyatis, J. Kim, A. J. Miller, and D. J. Gauthier, Universal Model for the Turn-On Dynamics of Superconducting Nanowire Single-Photon Detectors, Phys. Rev. Appl. **12**, 034020 (2019).

[8] B. Dolgoshein *et al.*, Status report on silicon photomultiplier development and its applications, Nucl. Instrum. Methods Phys. Res. A **563**, 368 (2006).

[9] D. O'Connor, *Time-Correlated Single Photon Counting* (Academic press, New York, USA, 2012).

[10] D. Shin, A. Kirmani, V. K. Goyal, and J. H. Shapiro, Photon-efficient computational 3-D and reflectivity imaging with single-photon detectors, IEEE Trans. Comput. Imaging **1**, 112 (2015).

[11] D. Shin, F. Xu, D. Venkatraman, R. Lussana, F. Villa, F. Zappa, V. K. Goyal, F. N. Wong, and J. H. Shapiro, Photon-efficient imaging with a single-photon camera, Nat. Commun. **7**, 1 (2016).

[12] J. Rapp and V. K. Goyal, A few photons among many: Unmixing signal and noise for photon-efficient active imaging, IEEE Trans. Comput. Imaging **3**, 445 (2017).

[13] D. B. Lindell, M. O'Toole, and G. Wetzstein, Single-photon 3D imaging with deep sensor fusion, ACM Trans. Graph. **37**, 113 (2018).

[14] J. Peng, Z. Xiong, X. Huang, Z.-P. Li, D. Liu, and F. Xu, in *European Conference on Computer Vision (ECCV)* (Springer, Glasgow, UK, 2020), p. 225.

[15] Z. Zang, D. Xiao, and D. D.-U. Li, Non-fusion time-resolved depth image reconstruction using a highly efficient neural network architecture, Opt. Express **29**, 19278 (2021).

[16] P. Rehain, Y. M. Sua, S. Zhu, I. Dickson, B. Muthuswamy, J. Ramanathan, A. Shahverdi, and Y.-P. Huang, Noise-tolerant single photon sensitive three-dimensional imager, Nat. Commun. **11**, 1 (2020).

[17] A. M. Weiner, Femtosecond pulse shaping using spatial light modulators, Rev. Sci. Instrum. **71**, 1929 (2000).

[18] A. Torralba and A. A. Efros, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Colorado Springs, CO, USA, 2011), p. 1521.

[19] K. Weiss, T. M. Khoshgoftaar, and D. Wang, A survey of transfer learning, J. Big Data **3**, 1 (2016).

[20] K. Tang, Y. Da Wang, J. McClure, C. Chen, P. Mostaghimi, and R. T. Armstrong, Generalizable Framework of Unpaired Domain Transfer and Deep Learning for the Processing of Real-Time Synchrotron-Based X-Ray Microcomputed Tomography Images of Complex Structures, Phys. Rev. Appl. **17**, 034048 (2022).

[21] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, A comprehensive survey on transfer learning, Proc. IEEE **109**, 43 (2020).

[22] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Salt Lake City, UT, USA, 2018), p. 3723.

[23] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, Domain-adversarial training of neural networks, J. Mach. Learn. Res. **17**, 2096 (2016).

[24] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Honolulu, HI, USA, 2017), p. 7167.

[25] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Salt Lake City, UT, USA, 2018), p. 7472.

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Las Vegas, NV, USA, 2016), p. 2818.

[27] R. Christoph and F. A. Pinz, in *Advances in Neural Information Processing Systems (NeurIPS)* (PMLR, Barcelona, Spain, 2016), p. 3468.

[28] G. Yao, Y. Chen, Y. Liu, X. Hu, and Y. Pan, Robust photon-efficient imaging using a pixel-wise residual shrinkage network, Opt. Express **30**, 18856 (2022).

[29] I. J. Myung, Tutorial on maximum likelihood estimation, J. Math. Psychol. **47**, 90 (2003).

[30] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga *et al.*, in *Advances in Neural Information Processing Systems (NeurIPS)* (PMLR, Vancouver, BC, Canada, 2019), p. 8024.

[31] D. P. Kingma and J. Ba, in *International Conference on Learning Representations (ICLR)* (OpenReview.net, San Diego, CA, USA, 2015).

[32] Code repository for deep domain adaptation photon-efficient imaging using STIN, https://github.com/ewellchen/DA-STIN.

[33] P. K. Nathan Silberman, Derek Hoiem, and R. Fergus, in *European Conference on Computer Vision (ECCV)* (Springer, Florence, Italy, 2012), p. 746.

[34] D. Scharstein and C. Pal, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE Computer Society, Minneapolis, Minnesota, USA, 2007), p. 1.