2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 23-27, 2022, Kyoto, Japan

Sim2Real Instance-Level Style Transfer for 6D Pose Estimation

Takuya Ikeda, Suomi Tanishige, Ayako Amma, Michael Sudano, Hervé Audren, and Koichi Nishiwaki*

***Abstract*— In recent years, synthetic data has been widely used in the training of 6D pose estimation networks, in part because it automatically provides perfect annotation at low cost. However, there are still non-trivial domain gaps, such as differences in textures/materials, between synthetic and real data. These gaps have a measurable impact on performance. To solve this problem, we introduce a simulation to reality (sim2real) instance-level style transfer for 6D pose estimation network training. Our approach transfers the style of target objects individually, from synthetic to real, without human intervention. This improves the quality of synthetic data for training pose estimation networks. We also propose a complete pipeline from data collection to the training of a pose estimation network and conduct extensive evaluation on a real-world robotic platform. Our evaluation shows significant improvement achieved by our method in both pose estimation performance and the realism of images adapted by the style transfer.**

2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-7927-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IROS47612.2022.9981878

I. Introduction

Determining the location and orientation (6D pose) of detected objects in the 3D world, is a critical ability for robots that are meant to manipulate objects. As the field progresses towards robot deployment in increasingly unstructured settings, such as human homes, the ability to train robust perception systems across a wide variety of situations has become a key bottleneck. Complicating matters, not only is there a large variety of household objects, but such objects can be transparent, shiny, or have any number of challenging characteristics. In this paper, we tackle RGB-based 6D pose estimation for these types of challenging objects.

Deep neural networks have made great progress in RGB-based 6D pose estimation [1], [2], [3]. However, a large amount of data collection and annotation is required in the training phase to achieve high performance. Moreover, since annotation of 6D object poses is more difficult than 2D image labeling, the labor required for these tasks is immense, especially when only RGB images are available [4], [5]. One way to overcome this difficulty is to leverage synthetic data; virtually infinite data and perfect annotations can be generated at low cost via rendering. However, although these features are attractive, there exists a distinct domain gap between synthetic and real data, resulting in sub-optimal network performance.

In the past few years, numerous methods have been proposed to solve the domain gap problem. One promising approach is the combination of domain randomized (DR) and photo-realistic data, as described in [6], [7]. While these approaches have shown great promise when high-quality 3D models are available, performance often degrades when using noisy 3D assets that are acquired in the wild. Even when high-quality models and renderers are available, there still exists a gap between simulation and real. To bridge this gap, domain adaptation techniques are promising, as they allow the conversion of synthetic images into realistic ones, utilizing real images without annotation. One popular approach here is the use of Generative Adversarial Networks (GANs). For example, Rojtberg *et al.* [8] leveraged CycleGAN [9] to transfer synthetic images to realistic ones for 6D pose estimation networks. However, as described in INIT [10], if the target domain is a complex scene containing multiple objects, serious inconsistencies will occur because these methods [9], [11], [12] focus on directly adapting a global style to the entire image.

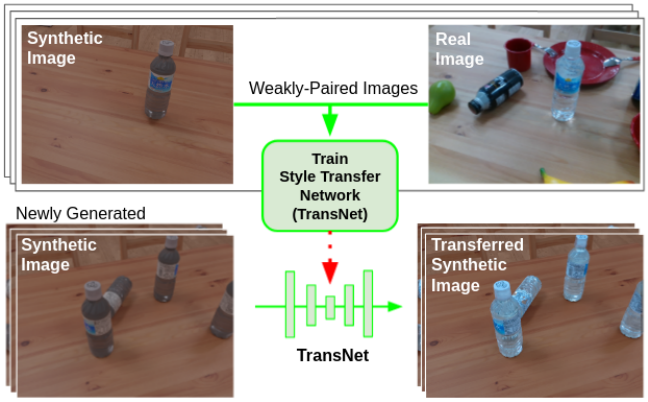

Fig. 1. Top: Training of the style transfer network using weakly-paired images between synthetic and real. Bottom: Style transfer from newly generated synthetic images to real ones, using the style transfer network.

Our key insight is that transferring the style of individual objects instead of the whole image style will result in more correspondences between the domains. Moreover, pose-level correspondences of multicolored objects are an important part of style transfer; correspondences of instance locations in 2D space only will result in appearance mismatches of texture-rich objects. For example household objects often have a very different style depending on the viewpoint. Thus, we propose a sim2real domain adaptation method for 6D pose estimation via instance-level style transfer that is capable of handling complex scenes without manual annotations. Our objective is to improve the capability of an off-the-shelf 6D pose estimation network by improving synthetic data quality. The proposed method is able to transfer multiple texture/material-rich object styles individually. Fig. 1 shows an overview of our sim2real instance-level style transfer.

* All authors are with the Woven Planet Holdings, Inc. 3 Chome-2-1 Nihonbashimuromachi, Chuo City, Tokyo 103-0022, Japan, [firstname.lastname]@woven-planet.global

To summarize, our key contributions are:

1) An automatic pipeline to produce weakly-paired images between synthetic and real for training sim2real style transfer networks as described in Sec. III-B.
2) A procedure for instance-level object style transfer from synthetic to realistic as described in Sec. III-C.

Our contributions improve the performance of a 6D pose estimation network as demonstrated in Sec. IV.

II. Related Work

A. Learning with Synthetic Data

Recently, synthetic data has been widely used in the training of 6D pose estimation systems because it naturally gives perfect annotations at a low cost. DOPE [6] have successfully trained 6D pose estimation network using only synthetic data which contain DR and photo-realistic rendered images. For DR, several methods [6], [13], [14], [15] are proposed. They randomize several parameters to get domain invariant features, such as poses of target objects and cameras, types of distractors, textures/materials of foreground and background, lighting settings, and sensor noise. While these methods show great results, it requires domain knowledge to find good parameters. With regards to creating photo-realistic images, there are also lots of research efforts [7], [16], [17], [18]. These sources prove that the qualities of the object model, such as textures/materials, and the renderers, including shadowing/lighting, are all important for sim2real. However, even with high quality 3D models, there always exist a difference between rendered and real images; this is the domain gap that degrades performance of pose estimation.

B. Domain Adaptation

Instead of trying to precisely model the textures/materials of target objects, domain adaptation methods seek to bridge these gaps using target domain's data [19], [20]. This approach can greatly reduce manual operations as mentioned previously. There are two primary methods of domain adaptation for 6D pose estimation networks without manual annotations. One way is to adapt the network to the target domain using geometric information. The reason for doing this is that geometry information is generally more domain invariant than RGB features and, therefore, a useful feature for pose estimation. For example, Self6D [21] compares rendered depth and real depth in order to minimize the error using a differentiable renderer. Rad *et al.* [22] also use depth information to enforce network invariant between synthetic and real. However, these methods are not available when depth information is lacking, as with transparent objects. The other method of domain adaptation for 6D pose estimation networks is a GAN-based approach that can be applied using only RGB information. CycleGAN [9] can translate unpaired images bidirectionally between two domains. For example, Rojtberg *et al.* [8] leverages the CycleGAN [9] to get a mapping between synthetic and real domain for pose estimation. Moreover, DRIT [12] and MUNIT [11] can handle multiple latent spaces for image-to-image translation. For example, they are able to disentangle style and content. However, if the target scene is complicated, for example containing multiple colorful objects, serious inconsistencies can occur. This is because these methods [9], [11], [12] focus on directly adapting a global style to the entire image. Richter *et al.* [23] translates synthetic images into realistic ones by using lots of intermediate rendering features for training. They show moderate improvements in performance by operating at the class level. By contrast, our approach is more targeted since we focus on individual instances and is simpler since we only require instance masks for synthetic labels. INIT [10] achieves instance-level translation, but it requires 2D bounding box annotations of real data, while we don't require any manual annotations.

III. Method

Our objective is to amplify the performance of 6D pose estimation networks (PoseNets) via style transfer, using synthetic and real data without manual annotations. We hypothesize that less data is required for the training of style transfer networks (TransNets) than those for PoseNets. We first collect and use minimal real data for TransNet training. Then, transfer a large number of synthetic images to more realistic ones, for PoseNet training. Our approach improves TransNet performance by providing data in which pose and instance information coincides between synthetic and real domains. Since ground truth labels are only available on synthetic data, TransNet must maintain the object poses in the image during the transfer. Our approach explicitly adapts the appearance of individual instances while keeping their poses in the scene.

To achieve the goal, our entire workflow is structured as shown in Fig. 2. First, we collect real images with a robot and generate synthetic images corresponding to these real images by using a PoseNet trained on the synthetic images only. Then, TransNet is trained based on these image pairs in an unsupervised manner. Finally, we train the PoseNet using the synthetic images adapted with the style transfer. In this section, we describe the key steps in detail. We assume that 3D models of the environment and target objects are given, and the camera pose in the global frame is provided by the robot system.

A. Real Data Collection by Robot

Real images are required for our domain adaptation method, although annotations are not. We use the robot for data collection in the target environment, which is originally designed for carrying out services in the environment. At the same time, the poses of the camera in the environment coordinate system are stored. They are used for rendering images which correspond to the collected real images in later steps. During the data collection, the robot captures the same target area from multiple viewpoints. This constraint is introduced in order to decrease detection mismatches in later steps. In our case, we utilized a gantry type robot installed on the ceiling of a room as shown in Fig. 2 (0).

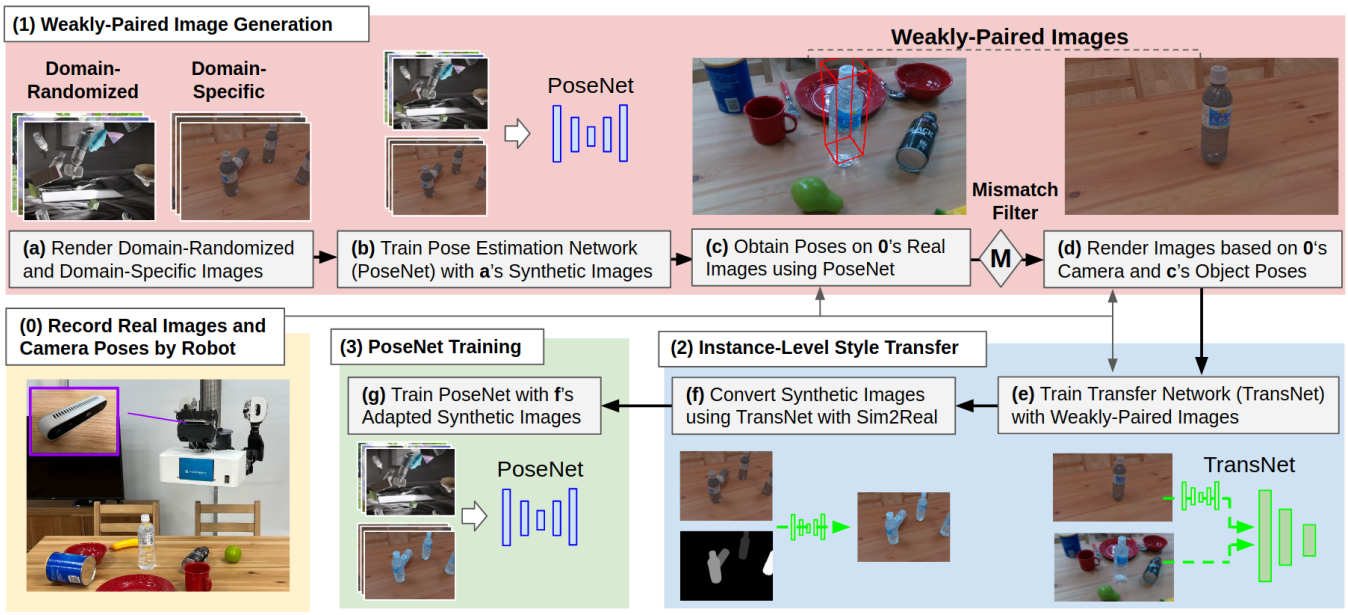

Fig. 2. **The workflow of the proposed domain adaptation and training method:** (1) Weakly-paired images are generated by using a robot and a 6D pose estimation network trained on synthetic images only. To generate these images properly, a mismatch filter is utilized as described in Sec. III-B. (2) Then, a style transfer network is trained based on the weakly-paired images, and convert synthetic images via instance-level style transfer. (3) Lastly, the 6D pose estimation network is trained using the adapted synthetic images.

B. *Weakly-Paired Images Generation*

Our goal here is to render scenes including the target objects and environment, which look similar to our collected real images, as shown in Fig. 2 (c)(d). We call these *weakly-paired images*. Since we already have camera poses and 3D models, the only missing information needed for rendering is the 6D pose of the target objects. To get these poses, we train the PoseNet using only synthetic data. Our approach can accept off-the-shelf RGB-based pose estimation networks. To obtain good performance, we generate two types of synthetic dataset like DOPE [6]. One is the Domain-Randomized dataset generated under the condition of random-placement sampling without considering gravitational effect together with random background, distractors, and lights (see Fig. 2 (a)); this aims to cover all the poses uniformly and realize robustness in variable conditions. The other one is the Domain-Specific dataset generated under the conditions of placing target objects considering gravitational effect in the 3D model of the target environment (see Fig. 2 (a)); this aims to cover the context of the target domain with the given models. Although the PoseNet trained with these datasets produces plausible results on real images, we observed that it does not detect a consistent number of instances across viewpoints of a single scene. This detection mismatch problem causes incorrectly generated weakly-paired images: some object instances are rendered in the wrong pose or not at all. Since images are taken from various viewpoints in a shared scene (see Sec. III-A) , they should contain the same number of detections. Thus, we eliminate images that do not match that number. This is called *mismatch filter*. We first run inference on every view i in a scene and compute the number of detections for a given object, n_i in a range of distances (in our case, from 0.3m to 1.5m). We then compute which value of n occurs most frequently, $\bar{n}$, and only use the views where $n_i = \bar{n}$ for further process. For example, if inferred results have one instance in 90% of images, two instances in 9%, and zero instances in 1%, we eliminate the latter 10% of images. Finally, we render synthetic images based on the estimated poses and obtain weakly-paired images.

C. *Instance-level Style Transfer*

Style transfer must preserve the shape and texture details of each target object. This is important since these features are entangled with the object's 6D pose. In this section, we show how to transfer the target object's style from synthetic to real images, using weakly-paired images.

Our TransNet is inspired from single image translations as described in CUT [24]. This method partially satisfies the above requirements by controlling the patch size for generator and discriminator. However, to solve the proper mapping from source to target, they assume that the images have similar contents in both domains (see Fig. 9 in [24]). While these assumptions allow the method to work, it is difficult to satisfy these in general indoor scenes. On the other hand, because weakly-paired images already contain similar contents around the object area between sim and real, we leverage these in order to eliminate the need for this assumption (see Fig. 3). The main difference between the original and our proposed algorithm is how to process the input and output image. The details are described below.

For TransNet training, we first crop both the synthetic and real images with the 2D bounding box of the target object area, using synthetic mask labels. In our case, the

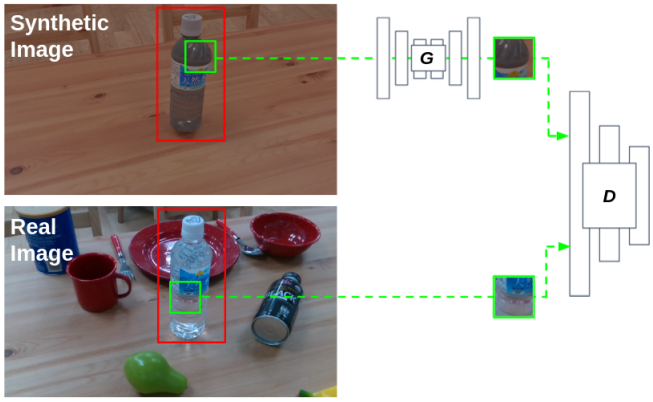

Fig. 3. Training of an instance-level style transfer network on weakly-paired images. G represents the style-mapping function from synthetic to real. D represents the discriminator for synthetic and real.

input image size is 640 x 480 and we extract a cropped area 1.2 times larger than the 2D bounding box. Then, we randomly scale the cropped images between 170 to 512 on the longest axis while keeping the aspect ratio. Lastly, we randomly sample fixed-size patches of 64 x 64 from the cropped-and-scaled image. This sampling is carried out independently for synthetic and real images. The patches are used as the network input (see Fig. 3). Using small patches of the target area allows us to preserve structure which is key of style transfer for PoseNet. While the original image transformation [24] includes random flipping, we have removed it. This is because asymmetrical information, such as labels and lettering, should be kept in our case. In addition, although the original method extracts input information from only a single image on each domain, we can use multiple images since all images are weakly-paired. Instead of a single *image* translation, our network performs a single *object style* translation. In other words, we train one TransNet for each object class, using multiple images.

The network architecture and loss function are the same as the single image translation [25]. The loss function is defined as:

$$L_G = L_{patchNCE}(G, H, X) + L_{patchNCE}(G, H, Y) + L_{GAN}(G, D, X, Y) + L_1(G, Y)$$
$$L_D = L_{GAN}(G, D, X, Y) + R_1(D, Y)$$

where X and Y represent the synthetic and real images, respectively; G represents the style-mapping function from synthetic to real; D is the style discriminator for synthetic and real; and H is a two-layer multilayer perceptron (MLP). The Generator and Discriminator are based on StyleGAN2 [26]. The Generator loss function is a combination of PatchNCE and non-saturating GAN Loss [27] with L1 regularization. The Discriminator loss is non-saturating GAN Loss with R1 gradient penalty stabilization [26], [28]. For more detail, please see the original paper [24]'s appendix B.

For style transfer with sim2real, we would like to preserve the object shape without losing details. Therefore, we carry

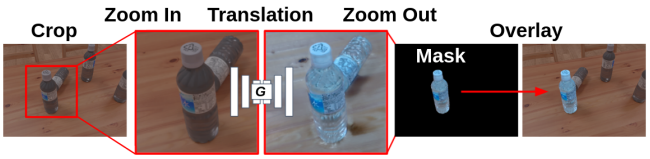

Fig. 4. The process of individual object style transfer

out the following pre-process and post-process as shown in Fig. 4. Before translation, we crop the input synthetic image with a 2D square bounding box and enlarge the image, in our case we scaled it to 512 x 512. After this process, we transfer the enlarged synthetic image and re-scale the image to its original size. This operation tends to result in less blurry images after translation (see appendix A). Lastly, we crop it with the masked area and overlay the masked image to the input image. This operation has the benefit of keeping the object silhouette. The following algorithm 1 presents the entire inference flow for a single image:

Algorithm 1: Inference flow for a single image

Require: image, network weights for target objects

$x \leftarrow image$
for $i \leftarrow$ class in image **do**
  for $j \leftarrow$ instance in class **do**
    $x_j \leftarrow \texttt{Crop}(image)$
    $x_j \leftarrow \texttt{Resize}(x_j)$
    $x_j \leftarrow \texttt{Infer}_i(x_j)$
    $x_j \leftarrow \texttt{Resize}(x_j)$
    $x_j \leftarrow \texttt{Mask}(x_j)$
    $x \leftarrow \texttt{Overlay}(x, x_j)$
  end for
end for
return x

Firstly, we load an input image to be used as the background. Then, we conduct the style transfer for each instance using the corresponding TransNet. Lastly, all of the transferred instance images are overlaid on the background image.

D. Training a 6D Pose Estimation Network

After training a TransNet for each object, you can transfer synthetic images that contains any number and any combination of target objects in one scene. For example, you can transfer an image which contains 5 water bottles, even if you trained the TransNet with only 1 water bottle in a scene, as shown in Fig. 1. Once the synthetic data has been adapted by our TransNet, the PoseNet is trained.

IV. Experiments

In this section, we quantitatively evaluate the effect of sim2real adaptation using our TransNets on the performance of PoseNets. Quantitative and qualitative comparison against several popular style transfer networks is also conducted.

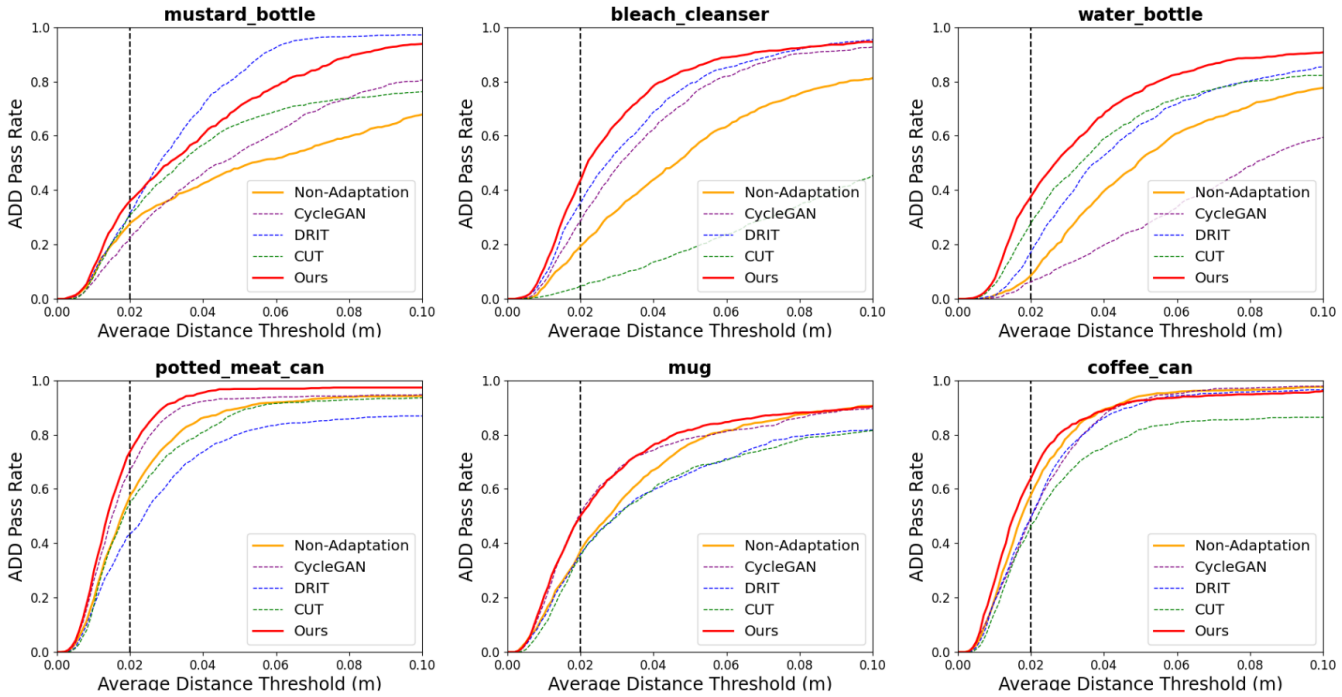

Fig. 5. Average distance threshold curves of non-adaptation, our adaptation method, and other methods including CycleGAN [9], DRIT [12], CUT [24].

TABLE I

ADD PASS RATE OF 2CM THRESHOLD AND SCORE OF AREA UNDER THE CURVE (AUC) FOR 6 HOUSEHOLD OBJECTS

Data / Object	mustard_bottle		potted_meat_can		bleach_cleanser		mug		water_bottle		coffee_can	
	ADD	AUC	ADD	AUC	ADD	AUC	ADD	AUC	ADD	AUC	ADD	AUC
Non-adaptation	0.280	42.74	0.579	74.50	0.196	48.32	0.377	63.39	0.089	43.62	0.584	76.99
CycleGAN [9]	0.227	48.67	0.674	78.38	0.290	61.51	**0.512**	66.97	0.063	27.41	0.503	75.19
DRIT [12]	0.321	**67.77**	0.437	65.63	0.357	65.12	0.364	57.16	0.176	52.64	0.499	74.56
CUT [24]	0.312	53.14	0.554	72.59	0.049	19.57	0.358	56.41	0.283	55.63	0.469	66.49
Ours	**0.363**	61.61	**0.742**	**81.75**	**0.442**	**69.33**	0.505	**68.42**	**0.381**	**63.70**	**0.645**	**77.65**

Fig. 6. **Target environment and objects:** The mustard_bottle and potted_meat_can are target objects in Dataset1. The bleach_cleanser and mug are targets in Dataset2. The water_bottle and coffee_can are targets in Dataset3. All other objects are distractors.

A. Experimental environment

Since we are targeting indoor environments in the real world, we evaluated our approach on a dining table in a home, as shown in Fig. 6. Our robot is installed on an X-Y gantry, above the table, and is able to access various viewing angles of a given object on the tabletop. For our target objects, we selected two sets of household objects. The first set is the popular *YCB Object Set* [29]. We picked 4 objects from this item set, representing a variety of colors and shapes: 006_mustard_bottle, 010_potted_meat_can, 021_bleach_cleanser, and 025_mug. Since household objects often contain transparent and shiny objects, we aim to evaluate our approach using objects with those features. Unfortunately, the YCB objects only contain base color information, and lack shading data. Thus, we have added two Japanese household object models, containing physically realistic and challenging material properties: a transparent water_bottle and a shiny coffee_can, as shown in Fig. 6.

B. Datasets and Metrics

We created three real-world datasets for our experiments, as shown in Fig. 6. Each dataset contains two target objects plus distractors (target objects: mustard_bottle / potted_meat_can, bleach_cleanser / mug, and water_bottle / coffee_can). Target objects were randomly placed on the table. For each dataset, 1k RGB images were gathered at a range of distances from 0.3m to 1.5m, covering common use-cases. The images were captured by an *Intel RealSense D415*

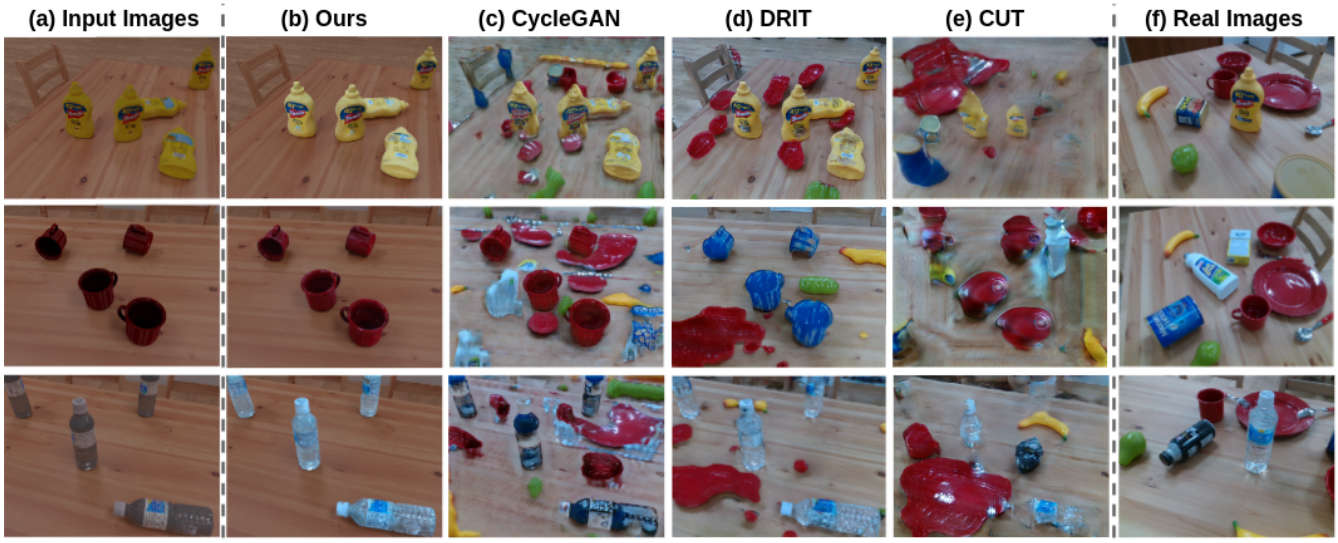

Fig. 7. **Style Transfer Results**: We compare ours to other transfer methods including CycleGAN [9], DRIT [12], CUT [24].

mounted on the head of our robot, as shown in Fig. 2 (0).

We evaluated 6D pose estimation performance using the Average Distance of Distinguishable Model Points (ADD) metric in [3], [30]: the average distance of all model points between estimation and ground truth. Ground truth labels, used only for evaluation, were obtained via manual labeling.

C. Training

We selected PVNet [25] as a PoseNet for our experiments. For the training, we used two types of synthetic datasets, as described in Sec. III-B. These datasets contain a total of 5k images of each target object as shown in Fig. 2 (a):

- 2.5k Domain-Randomized images.
- 2.5k Domain-Specific images.

Each image contains a maximum of 5 instances from the same class. We trained 6 PoseNets individually, one for each object, from scratch. Fig. 7 (a) shows samples of the Domain-Specific images and Fig. 7 (b) shows the results of our domain adaptation via TransNet. Since TransNet performs domain-specific adaptation, we only adapted Domain-Specific images; the Domain-Randomized dataset was used without domain adaptation. For the PoseNet training, we set the epoch and batch sizes to 50 and 2 respectively.

For TransNet, we updated the data processing parts of single image translation [24] to the one described in Sec. III-C. We gathered 1k real images for each dataset as described in the Sec. IV-B, and then sifted them using our mismatch filter. The remainder (the average: 86.5% in our case) was used for TransNet training, for which we set the epoch and batch sizes to 16 and 16 respectively.

All networks were implemented using *PyTorch*. For rendering synthetic images, we used *Blender Cycles*.

D. Evaluation Results and Comparison

To exhibit how PoseNet is adapted to a target domain via our TransNet, we performed a comparison between non-adaptation and our adaptation, using the ADD metric. Moreover, we compared these and 3 other methods: CycleGAN [9], DRIT [12], and CUT [24]. To ensure a fair comparison we used the same data in all cases and trained the same PoseNet on the adapted images. Although the compared methods can accept unpaired images, we trained them using the same weakly paired images used for our TransNet. Fig. 5 shows the resulting average distance threshold curves: the orange solid curve shows the accuracy of PoseNet trained on the synthetic data without adaptation, the red solid curve shows with our adaptation, the purple dashed curve shows with CycleGAN [9], the blue dashed curve shows with DRIT [12], the green dashed curve shows with CUT [24]. The vertical dotted line indicates a 2cm threshold. This level of accuracy is necessary in order to accomplish proficient object grasping using a robot, as described in DOPE [6].

The results in Fig. 5 and Table I indicate that only our TransNets improved the performance for all target objects compared to the non-adaptation. In contrast, the all other works show uneven results. For example, while DRIT [12] successfully improved the AUC score for mustard_bottle, the scores for potted_meat_can, mug and coffee_can are lower than non-adaptation. They also show that our approach is applicable not only to opaque objects, but also transparent.

Fig. 7 presents a qualitative comparison between our method and the above 3 methods. The competing methods contain collapsing shapes (the second row of Fig. 7 (e)), style mismatches (the second row of Fig. 7 (d)), and visible artifacts around objects (the third row of Fig. 7 (c)). The quality of these adapted images is therefore unreliable depending on scene complexity. We hypothesize that these effects, for example, the style mismatch of DRIT on the mug (the second row of Fig. 7 (d)), are responsible for its poor 6D pose estimation performance on that object. In contrast, our method is able to generate images stably without these effects.

E. Discussion

We have noticed that our approach benefits where there is a significant gap between sim and real. On one hand, the texture/material quality of the 3D-modeled coffee_can shown in Fig. 8 (a) is already excellent. As the domain gap is small, the improvement is trivial (an increased score of AUC: 0.66). On the other hand, big gains are had for the two objects that have large domain gaps in modeling: water_bottle (with an increased score of AUC: 20.08) and bleach_cleanser (with an increased score of AUC: 21.01). As shown in Fig. 8 (b), the synthetic image of water_bottle is very different from the real one, mainly due to the complexity of modeling and rendering transparent materials. In the case of bleach_cleanser, our real object does not have a red label that's present on the 3D model due to a change in packaging as shown in Fig. 8 (c). It is quite common for the appearance of household objects (labeling details, textures, text, etc.) to change. For example, seasonal accents might be introduced, promotional labels added, or expiration date placement and details changed. Since our approach is capable of bridging these gaps successfully, it will enable wider usage of sim2real.

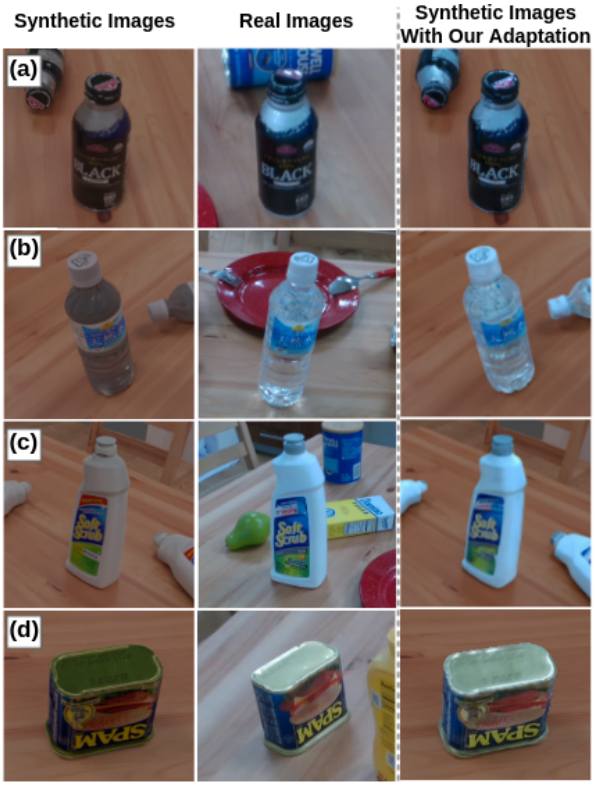

Fig. 8. **Sim2Real Gaps**: (a) The appearance gap is small. (b) Synthetic image is sub-optimal due to transparent material settings. (c) The model is different from the real one which only has the red at the top of its front label. (d) The gap is large due to shiny material settings. Our approach is capable of bridging these gaps.

V. Conclusions

We showed a sim2real domain adaptation technique for 6D pose estimation, without any manual annotations. Our approach was able to automatically generate weakly-paired synthetic and real datasets that are traditionally difficult to obtain. The proposed method was able to translate the styles of texture/material-rich objects from synthetic to realistic in cluttered indoor scenes, as shown in Fig. 6 and Fig. 8. Moreover, evaluation results proved the proposed method amplifies the performance of a 6D pose estimation network for all target objects. As such, when 3D models and camera poses are available, one can automatically adapt the network to a target environment in the real world at low cost.

In this paper, the style transfer networks have to be trained for each class individually and focus on foreground style transfer only. The next step will be to perform foreground and background style transfer using a single network. Also, we will carry out further evaluations, such as how the amount and quality of data affect its performance. While we assumed 3D models are available, that is not always the case. Therefore, in future works, we would like to employ real2sim techniques that automatically produce 3D models from real images, such as photogrammetry and NeRF [31].

Appendix

A. Scaling Operation for TransNet Inference

The scaling operation as described in Sec. III-C preserves texture details, as shown in Fig. 9. Indeed, the text of the label in Fig. 9 (b) without the scaling operation is blurry.

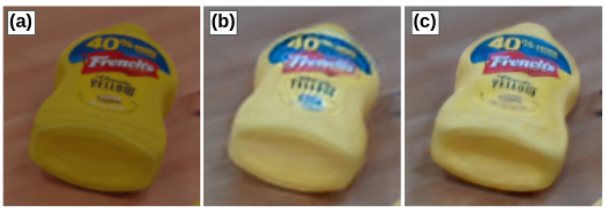

Fig. 9. (a) Input image. (b) Transferred image without scale-up. (c) Transferred image with scale-up.

B. Style Transfer for Multi-Class

Once you train individual TransNets for each target object, you can transfer synthetic images which contain multiple classes, as shown in Fig. 10. Since training is done independently for each object, style transfer can be completed without training data containing all the target objects in the same scene. At inference time, individual instances of each class are cropped out, as shown in Fig. 4, and inference is performed with the TransNet corresponding to that class.

References

[1] M. Rad and V. Lepetit, "Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3828–3836.

[2] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1521–1529.

[3] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.

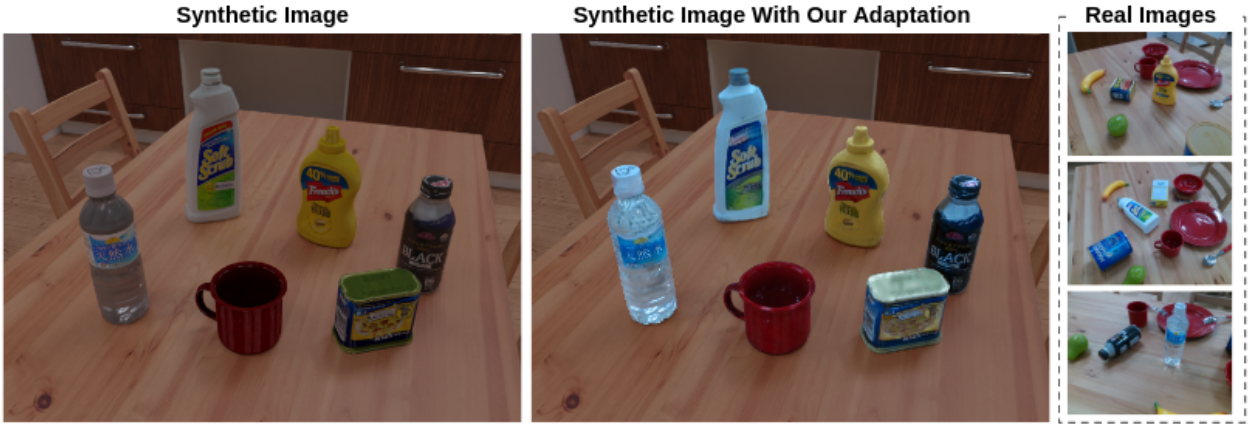

Fig. 10. Style transfer results for multi-class objects in a shared scene

[4] S. Hinterstoisser, O. Pauly, H. Heibel, M. Martina, and M. Bokeloh, "An annotation saved is an annotation earned: Using fully synthetic training for object detection," 2019.

[5] Z. Yang, X. Yu, and Y. Yang, "Dsc-posenet: Learning 6dof object pose estimation via dual-scale consistency," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3907–3916.

[6] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018.

[7] R. Alghonaim and E. Johns, "Benchmarking domain randomisation for visual sim-to-real transfer," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 12 802–12 808.

[8] P. Rojtberg, T. Pöllabauer, and A. Kuijper, "Style-transfer GANs for bridging the domain gap in synthetic pose estimator training," in *2020 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, Dec. 2020, pp. 188–195.

[9] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[10] Z. Shen, M. Huang, J. Shi, X. Xue, and T. S. Huang, "Towards instance-level image-to-image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3683–3692.

[11] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 172–189.

[12] H.-Y. Lee, H.-Y. Tseng, Q. Mao, J.-B. Huang, Y.-D. Lu, M. Singh, and M.-H. Yang, "Drit++: Diverse image-to-image translation via disentangled representations," *International Journal of Computer Vision*, vol. 128, no. 10, pp. 2402–2417, 2020.

[13] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[14] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 969–977.

[15] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3d orientation learning for 6d object detection from rgb images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 699–715.

[16] M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and H. Katam, "Blenderproc," *arXiv preprint arXiv:1911.01911*, 2019.

[17] M. Roberts, J. Ramapuram, A. Ranjan, A. Kumar, M. A. Bautista, N. Paczan, R. Webb, and J. M. Susskind, "Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding," in *International Conference on Computer Vision (ICCV) 2021*, 2021.

[18] M. Schwarz and S. Behnke, "Stillleben: Realistic scene synthesis for deep learning in robotics," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 502–10 508.

[19] S. Bi, K. Sunkavalli, F. Perazzi, E. Shechtman, V. G. Kim, and R. Ramamoorthi, "Deep cg2real: Synthetic-to-real translation via image disentanglement," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2730–2739.

[20] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim, "Image to image translation for domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4500–4509.

[21] G. Wang, F. Manhardt, J. Shao, X. Ji, N. Navab, and F. Tombari, "Self6d: Self-supervised monocular 6d object pose estimation," in *European Conference on Computer Vision*. Springer, 2020, pp. 108–125.

[22] M. Rad, M. Oberweger, and V. Lepetit, "Domain transfer for 3D pose estimation from color images without manual annotations," in *Computer Vision – ACCV 2018*. Springer International Publishing, 2019, pp. 69–84.

[23] S. R. Richter, H. A. AlHaija, and V. Koltun, "Enhancing photorealism enhancement," *arXiv preprint arXiv:2105.04619*, 2021.

[24] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, "Contrastive learning for unpaired Image-to-Image translation," in *Computer Vision – ECCV 2020*. Springer International Publishing, 2020, pp. 319–345.

[25] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "Pvnet: Pixel-wise voting network for 6dof pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4561–4570.

[26] T. Karras, S. Laine, and T. Aila, "A Style-Based generator architecture for generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4217–4228, Dec. 2021.

[27] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[28] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for GANs do actually converge?" in *International conference on machine learning*, 2018, pp. 3481–3490.

[29] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols," *arXiv preprint arXiv:1502.03143*, 2015.

[30] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of Texture-Less 3D objects in heavily cluttered scenes," in *Computer Vision – ACCV 2012*. Springer Berlin Heidelberg, 2013, pp. 548–562.

[31] B. Mildenhall, P. Srinivasan, M. Tancik, J. T. Barron, and R. R. R. Ng, "Representing scenes as neural radiance fields for view synthesis," in *Proc. of European Conference on Computer Vision, Virtual*, 2020.