Pixel-Level Domain Adaptation for Real-to-Sim Object Pose Estimation

Kun Qian, *Member, IEEE*, Yanhui Duan, Chaomin Luo, *Senior Member, IEEE*, Yongqiang Zhao, Xingshuo Jing

***Abstract*—Transferring robotic grasping skills learned from a simulator to the real world is beneficial in reducing the cost of labeling. However, the models trained on synthetic data are brittle when being applied to real-world data due to the domain gap. In this paper, we propose CCM Pixel-DA, a novel real-to-sim approach to unsupervised domain adaptation for object pose estimations, which outperforms conventional domain adaptation methods in preserving structural information, semantic information, and object pose during the transfer. The pipeline decouples domain adaptation and pose estimation, which allows the CCM Pixel-DA method to be integrated into state-of-the-art object pose estimation networks. The proposed method is further integrated into a pipeline for robot grasping. Experimental results on a real-world robot grasping system validate that the system is capable of grasping real-world objects without object pose annotations in the real-world domain.**

***Index Terms*—Transfer learning, Domain adaptation, Generative adversarial networks, Object pose estimation, Robot grasping**

I. Introduction

DATA-DRIVEN object pose estimations are highly reliant on the number of training samples. With the rapid progress in synthetic image generation [1][2][3], using synthetic data as model input has become an alternative method for data-driven learning [4][5]. Unfortunately, models naively trained on synthetic data fail to be generalized to real images because of reality gaps [6].

A solution to this problem is unsupervised pixel-level domain adaptation (Pixel-DA)[7][8]. For example, [7] performs pixel-level alignment to translate the style of source data to a target domain. It aims to minimize the domain adversarial loss but unnecessarily preserves the content. The result is that crucial semantic information may be lost. To handle this issue, Cycle-consistency has been recently proposed in a GAN-based cross-domain image generation model [9]. However, the combination of Cycle-consistency and domain adversarial loss fails to meet the demand of transferring object pose estimation samples, because it lacks the shape and category constraints on a single sample while the samples affect each other during training.

Thus we consider this demand for preserving the object information during the conversion from the psychological perspective. The cognitive psychology studies [10][11] disclosed that the humans use visual attention conditioned on objects' geometric and spatial properties to generate motor signals. Thus the key to preserving the object is to keep the object unchanged during the conversion, which aims to keep the same visual attention for the robot grasping like a human [12]. It can be formalized via the use of an additional loss that penalizes large differences between source and generated images for foreground pixels only, called Content-consistency.

In this paper, we present a Pixel-DA method for unsupervised real-to-sim object pose estimation using unpaired samples. The proposed method follows the unsupervised learning framework, as it's independent of real-world object pose annotations. The core of our approach is the improvement of the Pixel-DA method [7] by combining Cycle-consistency, Content-consistency, and Mapping-consistency (CCM). Therefore, the proposed method is called CCM Pixel-DA. In other words, we have improved the CycleGAN network [9] by adding the content consistency to preserve the structural and semantic similarity during conversion.

In summary, the contributions of this work are as follows:

- We have proposed the CCM Pixel-DA model, a Pixel-Level domain adaptation (Pixel-DA) model based on Cycle-consistency, Content-consistency, and Mapping-consistency (CCM) for transferring pose estimation samples from the real-world domain to the simulation domain without any real-world labeled data. The model outperforms previous work on preserving structural information, semantic information, and object pose during the transfer.
- We have proposed a pipeline for robot grasping based on domain adaptation and data-driven pose estimation, which promotes robot grasping without human annotations and data collection in the real world.
- The experimental results on both dataset and a real-world UR5 robot have validated the effectiveness of our methods in object pose estimation as well as robot grasping. An unpaired simulated and real-world dataset is also collected for testing the real-to-sim grasping pipeline.

The rest of this paper is organized as follows: Section II presents related work on domain adaptation and object pose estimations. The overview of the method and the robotic grasping system is described in Section III. Then, Section IV describes the details of the proposed method. Experimental results of pixel-level domain adaptation and 6D object pose estimation are reported in Section V, and the robotic grasping experiments are depicted in Section VI, followed by a conclusion in Section VII.

Corresponding author: Kun Qian, Chaomin Luo.

K. Qian, Y. H. Duan, Y. Q. Zhao and X. S. Jing are with the School of Automation, Southeast University and the Key Laboratory of Measurement and Control of CSE, Ministry of Education, No.2, Sipailou, Nanjing 210096, China. E-mail: kqian@seu.edu.cn

Chaomin Luo is with the Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39762 USA. E-mail:chaomin.luo@ece.msstate.edu

This article has been accepted for publication in IEEE Transactions on Cognitive and Developmental Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCDS.2023.3237502

II. Related work

Robot grasping methods can be roughly decomposed into two groups: object pose estimation methods and grasp pose detection methods. Grasp pose detection methods[13] extract the grasp pose candidates by directly sampling the scene point clouds, which lack shape sensitivity and global geometric information. Thus, the estimation method is inferior to the detection method in terms of grasping stability.

Data-driven object pose estimation methods predict the object poses from image data with deep neural networks in [14][15][16]. They formulate the pose estimation as a problem of estimating the rigid transformation containing 3D rotation and translation from the object coordinate system to the camera coordinate system, by training on a large amount of data. Recent pose estimation methods such as RePose [17], RNNPose [18] and ZebraPose [19] only employ RGB images, while FFB6D [20], PVN3D [21] and Densefusion [16] are based on RGB-D fusion.

A problem with the data-driven object pose estimation methods is that they rely heavily on the amount of human-labeled data. An appealing alternative is to use synthetic data [22][23][24][25] with pose annotations. However, the models trained purely on simulated data are often difficult to generalize well to the unseen real world due to a significant domain gap. In [26], domain randomization is utilized to bridge the gap for PVN3D-based pose estimation. In contrast, we focus on domain adaptation methods in this paper.

The solutions to domain adaptation can be divided into real-to-sim and sim-to-real methods. The sim-to-real methods in [27] learn to make synthetic data more realistic with an adversarial framework. Bousmalis *et al.* [4] presented the Grasp-GAN method which directly modifies synthetic images to make them more realistic and more useful for training a data-driven end-to-end vision-based grasping system. Jing *et al.* [28] transferred the grasping skills learned from simulated environments to the real world to complete the robot grasping.

Meanwhile, the real-to-sim method refers to generating synthetic data from real-world data, so that a model learned in simulators can deal with the synthetic data generated from real data. James *et al.* [29] proposed the model called RCAN mapping the heavily randomized simulation images with random textures and real images to a canonical (much simpler) rendered image. The pose estimation domain adaptation task is similar to the autonomous driving problem [30] in preserving the foreground while transferring the entire style without any annotations in the real-world domain. Therefore, we choose real-to-sim transfer in the object pose estimation inference stage. An in-depth description of the reasons will be explained in Section IV.

III. Overview

The pipeline of the pixel-level domain adaptation for real-to-sim 6D object pose estimation is composed of two models: real-to-sim Transfer Model and 6D Pose Estimation Model. These two models are trained separately in the training stage, as shown in Figure 1.

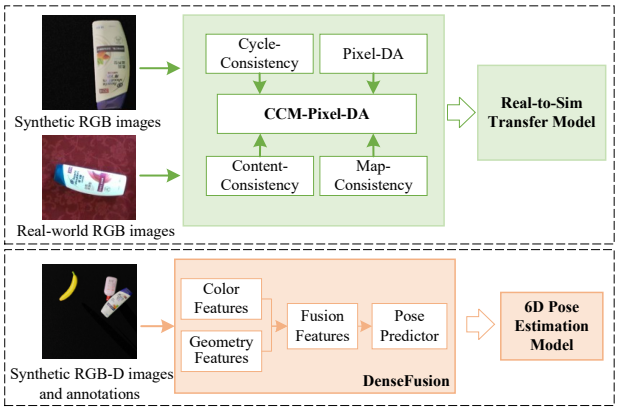

Fig. 1. The training pipeline of the Real-to-Sim Transfer model (top) and the 6D Pose Estimation model (bottom). Unpaired and unlabeled synthetic and real-world images are employed to train the Real-to-Sim Transfer model. Synthetic RGB-D images and annotations are used to train a 6D Pose Estimation model.

In the training stage, we initially utilize unpaired synthetic RGB images and real-world RGB images to train the Real-to-sim Transfer Model in Figure 1. The CCM Pixel-DA Network includes four modules: Pixel-DA, Cycle-Consistency, Content-Consistency, and Mapping-Consistency, as shown in Figure 1. Then we utilize the synthetic data including RGB images, depth images, object models, masks, and object poses, to train the 6D Pose Estimation Model in the simulation domain.

In our current implementation, we employ the end-to-end deep learning approach, DenseFusion [16], as the 6D pose estimator. It first detects object masks in RGB images using Mask R-CNN. Then it embeds and fuses the masked RGB values and object point clouds at a per-pixel level in combination with estimating the 6D pose of the object.

After training, the Real-to-Sim Transfer Model and 6D Pose Estimation Model are used in the inference stage, as illustrated in the blue block in Figure 2. The inference stage takes the real-world RGB-D images as input and transfers the real-world RGB images to the simulation domain. Then the object pose is estimated using the real-to-sim transferred RGB image and real depth image. Finally, the object pose is transformed into a grasp pose for guiding the robot to conduct a grasp.

The proposed method is further implemented as a robot grasping system, whose pipeline is depicted in Figure 2. Based on the 6D object pose estimation results ${}^{C}P_{object}$ derived from real-world RGB-D images, the system determines the best grasp pose ${}^{C}P_{grasp}$ in the camera coordinate system using an object superquadric modeling technique [31]. Then the grasp pose ${}^{B}P_{grasp}$ is calculated in the base coordinate system according to the eye-hand calibration results. The coordinate transform in the grasping pipeline is defined as follows:

$$ {}^{B}P_{grasp} = {}^{B}_{O}T\,{}^{O}P_{grasp}\,, \tag{1} $$

where ${}^{B}_{O}T$ refers to the transformation from the object coordinate system to the base coordinate system.

As determining grasp poses given object poses is out of the scope of this paper, any state-of-the-art model-based pose detection methods [32][33] can be utilized. More details of the system implementation will be described in Section VI.

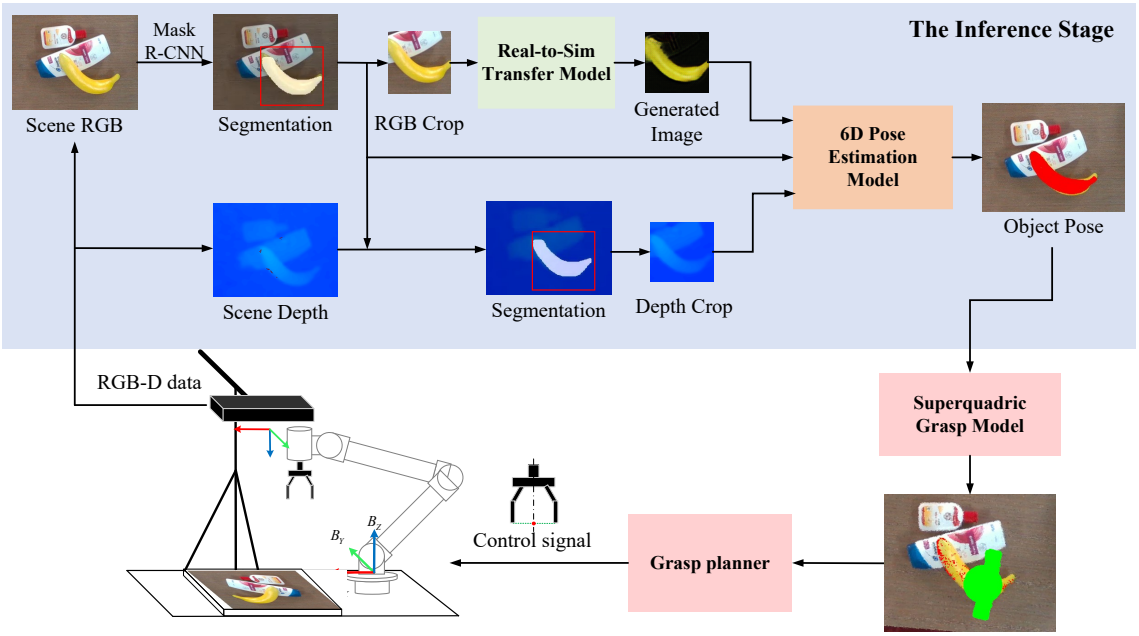

Fig. 2. The inference pipeline (blue block) and the overview of the robot grasping system. Firstly, the inference stage takes a real-world RGB-D image as input. Object segmentation is performed using Mask R-CNN to obtain an image crop. Then, the real-world RGB image crop is transferred to the simulation domain by the Real-to-Sim Transfer Model in the green block. Next, the object pose is estimated by the 6D Pose Estimation Model in the orange block, using the real-to-sim transferred RGB image crop and the real-world depth image crop. The proposed method is further implemented as a robot grasping system. Based on the 6D object pose derived from the inference stage, the best grasp pose is determined using an object superquadric modeling technique. Finally, the robot plans the grasping to pick up all objects in sequence.

IV. Pose Estimation Sample Transfer Based on Domain Adaptation

Sim-to-real transferred images are not sufficiently realistic because of the variations in image background and complex lighting conditions in real-world robotic grasping scenarios. In contrast, real-to-sim transfer quality can be assured because we can enforce the transfer module to generate images with consistent backgrounds and linear variations in light intensity. For these reasons, we consider the real-to-sim object pose estimation issue in this paper. In addition, since pose estimation samples in the simulation domain can be automatically annotated, our method can be independent of any labeled real-world pose estimation samples.

To meet the requirement of pose estimation, the layout and the poses of objects in the sample images are expected to be unchanged during the transfer. Therefore, our CCM Pixel-DA network extends the Pixel-DA method [7] by incorporating Cycle-consistency, Content-consistency, and Mapping-consistency. The architecture of the CCM Pixel-DA network is illustrated in Figure 3.

A. Pixel-DA Module

Let $X_S = \{x_s^i, y_s^i\}_{i=0}^{N_s}$ represent an unlabeled dataset of N_s samples from the source domain and let $X_T = \{x_t^i\}_{i=0}^{N_t}$ denote a labeled dataset of N_t samples from the target domain. Our pixel adaptation model consists of a generator $G_{S\to T}$ and a discriminator D_T. The generator maps a source domain image $x_s \in X_S$ and a noise vector $\mathrm{z} \sim p_z$ to an adapted image x_f. The discriminator outputs the likelihood d of a given image x being sampled from the target domain. The domain adversarial loss $\mathcal{L}_{\mathrm{GAN}}(G_{S\to T}, D_T, X_T, X_S)$ (see the parts with green outlines and green arrows in Figure 3) is defined as follows:

$$\begin{aligned}\mathcal{L}_{\mathrm{GAN}}(G_{S\to T}, D_T, X_T, X_S) \\ = \mathbb{E}_{x_t\sim X_T}\left[\log D_T(x_t)\right] \\ + \mathbb{E}_{x_s\sim X_S}\left[\log\left(1 - D_T\left(G\left(x_s\right)\right)\right)\right].\end{aligned} \tag{2}$$

The generator $G_{S\to T}$ is a convolutional neural network with residual connections as illustrated in Figure 4. The structure of the discriminator D_T is illustrated in Figure 5. In Figure 4 and Figure 5, CINR-k denotes a $n\times n$ Convolution-InstanceNorm-ReLU (CINR) layer [34] with stride k. RB $\times$ k denotes a residual block that contains two 3×3 convolution layers with the same number of filters on both layers.

B. Cycle-Consistency Module

With a separate Pixel-DA module, there is no way to guarantee that the generated samples preserve the structure or content of the original sample. The input image will be mapped to any random images in the target domain. Thus, the domain adversarial loss alone fails to guarantee an input x_i to the desired output y_i. In order to encourage the consistency of the generated sample $G_{S\to T}(x_s)$ and the input sample x_s, a cycle-consistency constraint is imposed on our adaptation module, referring to the parts with orange outlines and orange arrows in Figure 3. To be more specific, we introduce $G_{T\to S}$ as another mapping from the target domain to the source domain and define the corresponding domain adversarial loss as $\mathcal{L}_{\mathrm{GAN}}(G_{T\to S}, D_S, X_S, X_T)$. Thus, the sample from the source domain will be transformed to the target domain before back to the source domain to reproduce the original sample. In other words, we expect $G_{T\to S}(G_{S\to T}(x_s)) \approx x_s$ and

This article has been accepted for publication in IEEE Transactions on Cognitive and Developmental Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCDS.2023.3237502

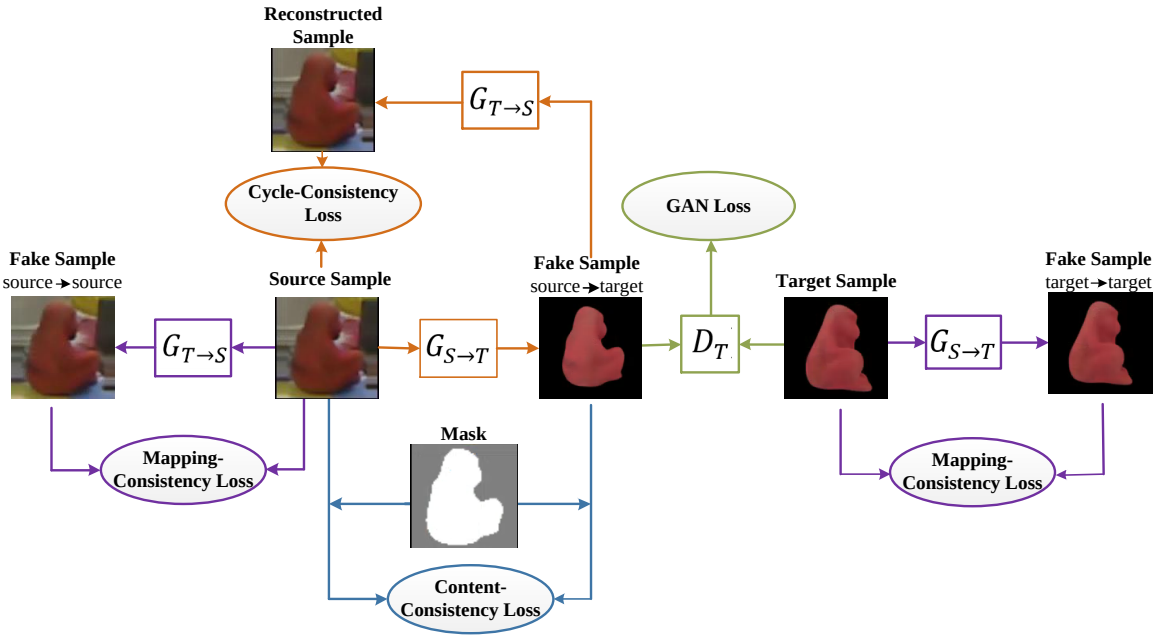

Fig. 3. The architecture of CCM Pixel-DA network. The CCM Pixel-DA network is built on the Pixel-DA domain adaptation method (the green part) and combines Cycle-consistency (the orange part), Content-consistency (the blue part) and Mapping-consistency (the purple part).

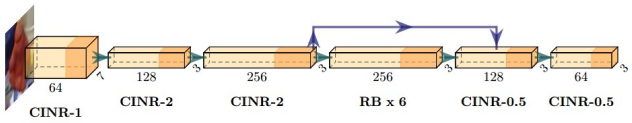

Fig. 4. The generator network in the Pixel-DA module. The generator is a convolutional neural network with residual connection. CINR-k denotes a $n \times n$ Convolution-InstanceNorm-ReLU (CINR) layer with stride k. $RB \times k$ denotes a residual block that contains two 3×3 convolution layers with the same number of filters on both layers.

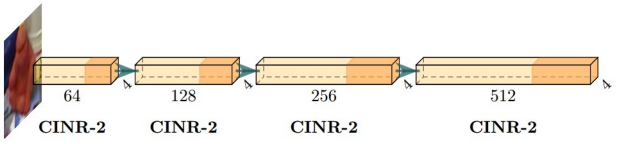

Fig. 5. The discriminator network in the Pixel-DA module. The CINR-k modules in the discriminator are similar to those in the generator.

$G_{S\to T}\left(G_{T\to S}\left(x_t\right)\right) \approx x_t$. This module enforces the cycle-consistency by minimizing the reconstruction error including $\mathcal{L}_{\text{cyc}}\left(G_{S\to T}, X_S\right)$ and $\mathcal{L}_{\text{cyc}}\left(G_{T\to S}, X_T\right)$:

$$\mathcal{L}_{\text{cyc}}(G_{S\to T}, X_S) = \mathbb{E}_{x_s\sim X_S}\left[\left\|G_{T\to S}(G_{S\to T}\left(x_s\right)) - x_s\right\|_1\right], \quad (3)$$

$$\mathcal{L}_{\text{cyc}}(G_{T\to S}, X_T) = \mathbb{E}_{x_t\sim X_T}\left[\left\|G_{S\to T}(G_{T\to S}\left(x_t\right)) - x_t\right\|_1\right]. \quad (4)$$

C. Content-Consistency Module

The cycle-consistency enforces the reconstruction error of all pixels. Object pose samples usually contain complex background and varied object pose estimation samples, the changes in the shape and pose of the objects in the samples will bring about cumulative errors in the prediction of object pose. Thus, we introduce the content-consistency module for calculating the masked Pairwise Mean Squared Error (PMSE) [35], referring to the parts with blue outlines and arrows in Figure 3. This loss penalizes the differences between pairs of pixels rather than absolute differences between inputs and outputs. Hence it allows the model to learn to reproduce the overall shape of the objects in the foreground.

In our case, we expect the images adapted from the source domain to have the same pose and category information but a different style in the foreground as the images from the target domain. Thereby, we extract the pose and category similarity as the object mask similarity. Given a binary mask, the masked-PMSE loss $\mathcal{L}_{\text{con}}\left(G_{S\to T}, X_S, M_S\right)$ and $\mathcal{L}_{\text{con}}\left(G_{T\to S}, X_T, M_T\right)$ are defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{con}}\left(G_{S\to T}, X_S, M_S\right) = \\ \mathbb{E}_{x_s\sim x_S}\left[\frac{1}{k_s}\left\|\left(x_s - G_{S\to T}\left(x_s\right)\right)\circ m_s\right\|_2^2\right. \\ \left. -\frac{1}{k_s^2}\left(\left(x_s - G_{S\to T}\left(x_s\right)\right)^\top m_s\right)^2\right],\end{aligned} \quad (5)$$

$$\begin{aligned}\mathcal{L}_{\text{con}}\left(G_{T\to S}, X_T, M_T\right) = \\ \mathbb{E}_{x_t\sim x_T}\left[\frac{1}{k_t}\left\|\left(x_t - G_{T\to S}\left(x_t\right)\right)\circ m_t\right\|_2^2\right. \\ \left. -\frac{1}{k_t^2}\left(\left(x_t - G_{T\to S}\left(x_t\right)\right)^\top m_t\right)^2\right],\end{aligned} \quad (6)$$

where k_s and k_t are the number of pixels in input k_s and k_t. m_s and m_t represent the corresponding mask of x_s and x_t, respectively. $\|\cdot\|_2^2$ is the squared L2-norm, and $\circ$ is the Hadamard product. This loss does not hinder the foreground from changing, but rather encourages the foreground to change in a consistent way.

Furthermore, the masks required for the real-world domain are obtained by self-supervised learning. A cross-domain semantic segmentation model is trained on abundant synthetic data before being applied to detect masks in the real-world domain.

D. Mapping-Consistency Module

In order to better preserve the color composition after conversion, we impose the mapping-consistency constraint on our model (see the parts with purple outlines and arrows in Figure 3). In particular, we regularize the generator $G_{S\to T}$ to be near an identity mapping when synthetic samples x_t of the target domain are provided as the input to the generator, which has also been applied to the cycle structure. The mapping consistency loss is defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{map}}\left(G_{S\to T}, G_{T\to S}\right) \\ = \mathbb{E}_{x_t\sim X_T}\left[\left\|G_{S\to T}\left(x_t\right)-x_t\right\|_1\right] \\ + \mathbb{E}_{x_s\sim X_S}\left[\left\|G_{T\to S}\left(x_s\right)-x_s\right\|_1\right].\end{aligned} \tag{7}$$

Taken together, these loss functions form the complete objective of our CCM Pixel-DA model:

$$\begin{aligned}\mathcal{L} =& \lambda_1\mathcal{L}_{\text{GAN}}\left(\boldsymbol{G}_{S\to T}, D_T, X_T, X_S\right) \\ &+ \lambda_2\mathcal{L}_{\text{GAN}}\left(G_{T\to S}, D_S, X_S, X_T\right) + \lambda_3\mathcal{L}_{cyc}\left(G_{S\to T}, X_S\right) \\ &+ \lambda_4\mathcal{L}_{cyc}\left(G_{T\to S}, X_T\right) + \lambda_5\mathcal{L}_{\text{con}}\left(G_{S\to T}, X_S, M_S\right) \\ &+ \lambda_6\mathcal{L}_{\text{con}}\left(G_{T\to S}, X_T, M_T\right) + \lambda_7\mathcal{L}_{\text{map}}\left(G_{S\to T}, G_{T\to S}\right),\end{aligned} \tag{8}$$

where $\lambda_1 \sim \lambda_7$ are the coefficients for each module. In our experiment, we take $\lambda_1 = \lambda_2 = 1, \lambda_3 = \lambda_4 = 10$, $\lambda_5 = \lambda_6 = 25, \lambda_7 = 5$ in experimental verification.

This ultimately corresponds to finding a target model f_T by solving the optimization problem:

$$\begin{aligned}f_T^* = \arg\min_{f_T} \min_{\substack{G_{S\to T}\\ G_{T\to S}}} \max_{D_S, D_T} \\ \mathcal{L}_{\text{CCM}-\text{Pixel}-\text{DA}}\left(f_T, X_S, X_T, Y_S, G_{S\to T}, G_{T\to S}, D_S, D_T\right).\end{aligned} \tag{9}$$

V. Experiments and Comparison Studies

In this section, we firstly evaluated our CCM Pixel-DA network in pixel-level domain adaptation. Then, an experiment of real-to-sim 6D pose estimation was conducted. The model training was undertaken on a computer with Intel i7 3.40GHz and NVIDIA GeForce GTX 2060.

A. Datasets

We evaluated our CCM Pixel-DA network on 6D Pose estimation dataset: LineMOD [36]. Furthermore, the Occlusion-LineMOD dataset was also used to test how well our method can deal with occlusions. The synthetic LineMOD dataset is built by uniformly sampling the 13 object models in LineMOD at different viewpoints using a simulated camera with the same intrinsic and extrinsic as the one used in LineMOD.

Fig. 6. An example in the real-world domain (left) and in the simulation domain (right). These examples are obtained by preprocessing the samples from LineMOD category 1. Then we feed these unpaired samples to train the CCM Pixel-DA network.

In the experiment of pixel-level domain adaptation, we chose and preprocessed the 13 sequences in LineMOD to build the dataset. The dataset contains 15275 samples in the real-world domain and 17069 samples in the simulation domain. We preprocessed the samples in both domains in light of the procedure as follows to highlight the content in the samples. Firstly, the bounding box was resized to a square, whose side length is taken as the longer side of the object bounding box. Then, the RGB and mask samples were cropped and resized to 128×128. An example of the preprocessed input samples is illustrated in Figure 6.

B. Evaluation Metrics

We evaluated the quality of the pixel-level image transfer with the Fréchet Inception Distance (FID) metric [37] and the Structural Similarity Metric (SSIM) [38]. Note that the better performance of a GAN model, the lower FID is supposed to be, whilst the SSIM will be 1 if the pair of images are identical [38]. We measured the accuracy of 6D pose estimation using the ADD(-S) metric [39] with 10% of the model diameter as the threshold.

C. Pixel-level Domain Adaptation Results

Three different methods as shown in Table II were compared. According to Section IV-C, the Content-consistency module essentially complements and modifies the functions of the Cycle-consistency module. Thus, the Content-consistency module was disabled until the Cycle-consistency loss converges to a certain degree δ, which will speed up the model convergence. Here, we set δ as 0.8. We kept the same training configurations for the three experiments listed. After 200 epochs, all losses of Method 1, 2, and 3 have converged.

For the identical inputs from the test datasets of 13 different categories, the generated samples $G_{S\to T}\left(x_s\right)$ converted by different methods are summarized in Figure 7. The qualitative evaluation results indicate that: a) Method 1 focuses more on the style transfer. The object in the generated samples has been changed, especially for "Apes" and "Phones"; b) Method 2 also lacks the ability to maintain the object poses, which will bring errors to the pose estimation. It is because the object shape and orientation have been changed after the conversion, such as "Apes", "Cans", "Cats" and "Drillers", as shown in Figure 7. In this case, the initial object pose annotation will not be exactly accurate, which brings errors to model training; c) Our method, $i.e.$, Method 3 preserves the object pose during conversion and the results of cross-domain transfer are satisfactory for object pose estimation.

For FID metrics, the quantitative results are outlined in Table I columns 2 to 5. We calculated the FID metrics of the real-world samples and the simulation samples as the benchmark. The input $(X_T, X_{S\to T})$ represents the FID metric for the samples in the target domain X_T and the generated samples $X_{T\to S}$ from the source domain to the target domain, which denotes the global transfer performance of the sample set. For SSIM metrics, the quantitative results are summarized in Table I columns 6 to 8. The benchmark is not provided here because the source domain samples are unpaired with the

TABLE I
REAL-TO-SIM DOMAIN ADAPTATION RESULTS IN FID AND SSIM METRICS.

	FID ↓				$\text{SSIM}_{\text{pair}}$ ↑		
	Benchmark	**Method 1**	**Method 2**	**Method 3**	**Method 1**	**Method 2**	**Method 3**
Objects/Input	(X_S, X_T)	$(X_T, X_{S\to T})$	$(X_T, X_{S\to T})$	$(X_T, X_{S\to T})$	$(X_S, X_{S\to T})$	$(X_S, X_{S\to T})$	$(X_S, X_{S\to T})$
Ape	286.2205	266.3891	66.3650	**50.1949**	0.4224	0.5021	**0.5062**
Benchvise	223.8625	218.0009	194.6081	**180.0514**	0.2257	0.2543	**0.2564**
Camera	217.2798	178.6218	143.6077	**138.7501**	0.3687	0.3763	**0.3877**
Can	198.4686	131.8329	64.4702	**62.7714**	0.2661	0.2865	**0.2901**
Cat	318.0181	62.2236	58.6680	**57.6020**	0.3729	0.3863	**0.3871**
Driller	285.7084	227.2660	197.9104	**151.8495**	0.3725	0.3748	**0.4081**
Duck	223.8036	42.5852	41.7439	**41.6722**	0.4786	0.4800	**0.4847**
Eggbox	296.4884	134.6908	109.2543	**75.8747**	0.2755	0.2938	**0.3067**
Glue	285.8064	179.0264	97.8512	**94.2455**	0.3217	0.3289	**0.3299**
Holepuncher	290.8866	138.7411	99.4927	**82.4587**	0.3811	0.3958	**0.4082**
Iron	229.9871	186.9853	157.9602	**141.8699**	0.2605	0.2635	**0.2658**
Lamp	245.2955	185.4244	169.2209	**153.1801**	0.2167	0.2281	**0.2317**
Phone	229.9269	226.6932	158.4866	**118.2970**	0.3190	0.3263	**0.3281**
Avg.	256.2887	165.4435	119.9645	**103.7552**	0.3293	0.3459	**0.3531**

Fig. 7. The pixel-level real-to-sim domain adaptation results of the 13 objects in LineMOD. We exhibit the real-world (source domain) samples in the first row. The real-to-sim results generated by Method 1, 2, 3 are shown in the second row to the fourth row, respectively. The nearest neighbors of the generated fake samples in the simulation (target domain) are given in the fifth row.

TABLE II
COMPARED METHODS IN THE ABLATION EXPERIMENTS.

Methods	**Illustration**
1	Pixel-DA [7]: Only Pixel-DA module.
2	CycleGAN [9]: Pixel-DA with Cycle-consistency module.
3	**Our method**: Pixel-DA with Cycle-consistency, Content-consistency and Mapping-consistency.

target domain samples. The input $(X_S, X_{S\to T})$ for $\text{SSIM}_{\text{pair}}$ depicts the average SSIM score of the paired source samples X_S and the generated samples $X_{S\to T}$. It illustrates the ability to preserve the structural consistency of objects.

In view of the quantitative and qualitative results, we can draw the conclusion that the cycle-consistency greatly improves the performance of the image style transfer. In addition, the content-consistency further improves the structure, content, and semantic similarity. As a result, Our CCM Pixel-DA network is qualified for real-to-sim object pose estimation.

D. 6D Pose Estimation Results

The experiments on 6D pose estimation followed the training and the inference pipeline in Section III. Since the poses of the generated samples were expected to remain unchanged during the transfer, the 6D pose annotations from the real-world domain were taken as the ground truth of the generated samples. Afterward, ADD(-S) scores were utilized as the

evaluation metric for pose estimation. Figure 9 visualizes some prediction examples output by our method.

TABLE III
THE ACCURACIES BASE ON ADD(-S) METRIC OF 6D POSE ESTIMATION IN LINEMOD AND OCCLUSION-LINEMOD.

	LineMOD			Occlusion-LineMOD		
Object	**DPOD**	**Self6D**	**Ours**	**DPOD**	**Self6D**	**Ours**
Ape	37.22	38.9	**47.25**	2.3	**3.7**	12.22
Benchvise	66.76	75.2	**90.72**	\	\	\
Camera	24.22	36.9	**86.03**	\	\	\
Can	52.57	65.6	**91.30**	4.0	**43.2**	42.96
Cat	32.36	57.9	**71.99**	1.2	18.7	**25.37**
Driller	66.60	67.0	**93.43**	10.5	32.5	**35.42**
Duck	26.12	19.6	**44.17**	7.2	14.4	**16.67**
Eggbox	73.35	**99.0**	96.52	4.4	**57.8**	50.32
Glue	74.96	**94.1**	88.52	12.9	**54.3**	49.86
Holepuncher	24.50	16.2	**37.70**	7.5	22.0	**25.67**
Iron	85.02	**77.9**	72.22	\	\	\
Lamp	57.26	68.2	**94.78**	\	\	\
Phone	29.08	50.1	**62.94**	\	\	\
Avg.	50	58.9	**73.66**	6.3	32.1	**32.43**

The left half of Table III reports the performance on the LineMOD dataset with other methods that only use synthetic data for training. DPOD solves the domain adaptation problem by freezing the first layers of the network trained on ImageNet and fine-tuning it with the synthetic data with domain randomization. Similar to our method, Self6D trains a pose estimation network using synthetic data. In the inference stage, the associated images of the real-world sample are generated by a neural rendering, *DIB-R*, to perform an alignment between real and rendered images w.r.t. the 6D pose.

According to the average ADD(-S) score, our method significantly outperforms DPOD [22] and Self6D [24] by 23.66% and 14.76%. However, for the object "Glue" with diverse projections from different views, the object "Iron" with complex textures, and the highly symmetrical object "Eggbox", our method is inferior to Self6D. This is because that DenseFusion is sensitive to the imperfect real-to-sim transfer, which can be observed from Table I that the FID score is non-zero, hence, the color embeddings extracted from the generated samples are slightly different from the color embeddings of the raw simulated samples. To understand the difference with Self6D, our pixel-level domain adaptation network can be trained independently, given any 6D pose estimation network pre-trained in simulation. However, Self6D needs two-stage training from scratch. In addition, our pixel-level domain adaptation network is a general real-to-sim method that can be applied to other end-to-end robotic manipulation tasks such as visual push and grasp [40].

The right half of Table III shows the evaluation of the robustness towards occlusion. Our method achieves the highest average accuracy of 32.43% on the Occlusion-LineMOD, in comparison with DPOD and Self6D. As the examples in Figure 8, the 1st and 2nd columns show the situations of partial occlusion where our method works well, and the 3rd and 4th columns show the case that the occlusion divides the object into two parts, in which our method has a limitation. Such an observation is understandable since the pose estimation on half of an object's RGB-D data is almost impractical.

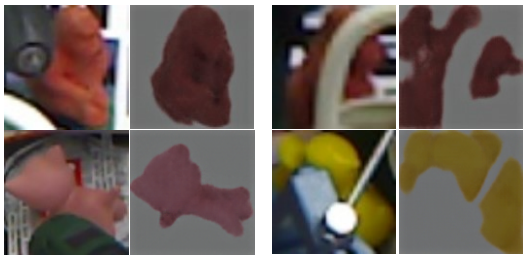

Fig. 8. Examples of image transfer results in Occlusion-LineMOD. Columns 1 to 2 show the cases with partial occlusion, where our method works well. Columns 3 to 4 show the cases with truncation, where pose estimation is difficult.

TABLE IV
THE ACCURACIES OF 6D POSE ESTIMATION IN HOMEBREWEDDB.

	Methods		
Object	**DPOD [22]**	**Self6D [24]**	**Ours**
Benchvise	52.9	72.1	**84.66**
Driller	37.8	65.1	**84.37**
Phone	7.3	41.8	**84.07**
Avg.	32.7	59.7	**84.37**

Table IV additionally shows the result in HomebrewedDB [41] based on ADD(-S) metric, which demonstrates the generality of our method. The results show our method outperforms significantly DPOD and Self-6D when the datasets with simpler scenarios and fewer occlusions, as shown in Column 7 of Figure 9.

VI. ROBOT GRASPING EXPERIMENTS

In this section, we aim to verify the viability and robustness of our grasping pipeline for pixel-level domain adaptation to real-to-sim 6D object pose estimations. In the grasping system, we employ a UR5 robotic arm equipped with a Robotiq 2-fingers gripper and RealSense D435 and we perform several online grasping experiments.

A. Datasets and Models

For obtaining synthetic data, we have developed a data generator using Blender. Our synthetic data generator supports rendering synthetic samples in BOP Format [36]. Furthermore, to guarantee the distribution of object poses widely enough to contain the real-world distribution, our data generator contains two modules: PhysicalPositioning and ObjectSampler. In PhysicalPositioning, CAD models are set with the specified height. In ObjectSampler, the object poses of one CAD model are uniformly sampled at certain intervals.

Then, we chose three object models including shampoo, banana, and lotion from Grasp-1 Billion [42] in the simulation domain to build our grasping dataset. We generated 1000 synthetic samples and annotations for each object, 400 of

This article has been accepted for publication in IEEE Transactions on Cognitive and Developmental Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCDS.2023.3237502

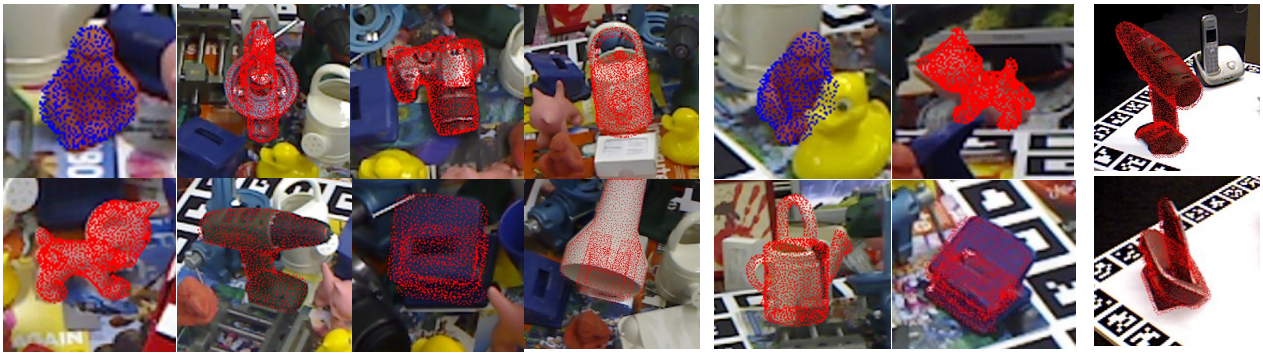

Fig. 9. Examples of object pose estimation results in LineMOD, Occlusion-LineMOD and HomebrewedDB. Object point cloud is transformed according to the predicted pose and then projected to the 2D image frame. Columns 1 to 4 depict the prediction in the LineMOD and Columns 5 to 6 show that in the Occlusion-LineMOD. Column 7 represents the pose prediction result in the HomebrewedDB.

them are generated with domain randomization of background for the cross-domain segmentation model, and 600 of them are generated with a black background for the CCM Pixel-DA network and DenseFusion. 300 samples of one object in different poses are generated using the ObjectSampler module with a black background and without occlusion. Another 300 samples of 3 objects are generated in random poses using the PhysicalPositiong module with 10% partial occlusion.

For obtaining real-world data, we collected these three object models and 225 samples without occlusion using RealSense D435 as shown in the right image in Figure 10. The real-world samples are augmented to 900 to train the CCM Pixel-DA network.

(a) (b) (c)

Fig. 10. Sample images from our dataset. (a) is an example of the synthetic samples generated by domain randomization, which are fed to train the cross-domain transferable Mask R-CNN. (b) shows an example of the synthetic samples with a consistent background, which are prepared for training the DenseFusion and CCM Pixel-DA network. (c) depicts an example of real-world samples, which are fed into the inference stage as well as preprocessed for training the CCM Pixel-DA network.

To prepare the domain adaptation dataset and train the domain adaptation model, we initially used the 3000 synthetic samples of 3 objects to train the Mask R-CNN network needed in DenseFusion. Previous work [43] proves that the trained Mask R-CNN has the ability for cross-domain semantic segmentation, in situations where the synthetic samples are generated with the same object model and camera configurations from the real world. Therefore, we utilized this cross-domain transferable semantic segmentation network to segment the 900 real-world sample images. The resulting 900 masked real-world images and the 1800 simulated images with rendered masks were pre-processed according to Section IV-A. This yields the dataset for training the domain adaptation network. After 100 epochs of training, the loss of the domain adaptation network tended to be stable.

To train the object pose estimation model, we used those 1800 synthetic samples with black backgrounds. During training for DenseFusion, we set the sample points on the model as 500 and the learning rate as 0.0002, which linearly decayed when test loss was less than 0.03. The iterative refinement was turned on if the test loss was less than 0.02. We kept the same training rate for the first 100 epochs and linearly reduced the rate to zero over the next 100 epochs. After 130 epochs of training, the model has converged.

B. Real-World Robot Grasping

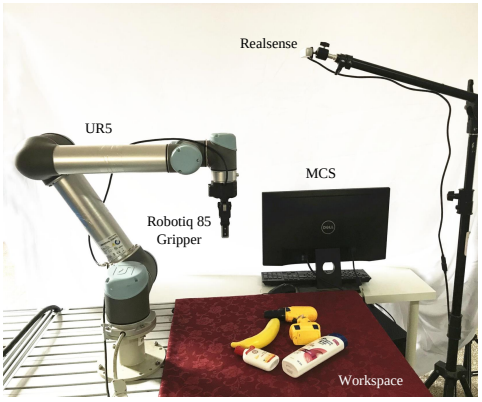

Fig. 11. The robotic grasping platform. It includes a UR5 robot with a Robotiq 85 gripper and a RGB-D sensor RealSense D435.

Using the models trained in Section V-B for the inference stage, the UR5 robot performed the grasping in a real-world setting as shown in Figure 11. The complete grasping pipeline, as illustrated in Figure 12, is as follows:

a) The raw image, as shown in Figure 12(a), was fed to the Mask R-CNN segmentation model to choose the object with the highest confidence and obtain the target image crop. Then, the image crop was transformed to the synthetic domain with the real-to-sim image transfer model, as shown in Figure 12(b).

b) The generated image, the depth image, and the object mask index were fed to DenseFusion for object 6D pose estimation. Figure 12(c) depicts the object poses by fitting the 3D model with red points to the scene point cloud.

c) The object pose was transformed to a grasp pose represented by the green gripper point sets based on the superquadric model, as shown in Figure 12(d).

d) The banana has been picked up successfully. Figures 12(e) and 12(f) show the snapshot when the gripper was closed and the object was picked up, respectively.

e) The remaining shampoo and lotion ware picked up in a similar way, as shown in Figures 12(g) to 12(l) and Figures 12(m) to 12(r).

TABLE V
THE STATISTICAL RESULT FOR GRASPING EXPERIMENTS.

Object	Grasping	Success	Success Rate%
Banana	20	20	100
Shampoo	20	18	90
SOD	20	17	85
Sum	60	55	**91.67**

We performed 60 robotic grasping trials with 3 objects in different poses. The statistical results of the grasping experiment are summarized in Table V, in which the average grasping success rate of our methods is 91.67%. The experiments validate that our method is effective for online robot grasping without any real annotations.

C. Discussion

Due to the complex scene background and object cluttering, we experimentally found that transferring a full image is difficult to maintain the shape of all objects in the scene. This is the reason why we propose object-level real-to-sim transfer by cropping an image, instead of transferring a full image. Therefore, our CCM Pixel-DA method is naturally integrated with DenseFusion, which accepts image crops as input.

However, this may hinder the direct and seamless integration of the proposed CCM Pixel-DA method with FFB6D [20] and PVN3D [21], which are also based on RGB-D data but require full images as input for pose estimation. While our method achieves an average accuracy of 73.66% on LineMOD, a recent work [26] that uses synthetic data and PVN3D reaches an average accuracy of 77.5%. This implies that replacing our pose estimation backbone with more recent networks is promising. We could improve the extendibility of the CCM Pixel-DA method by background subtraction so that the cropped images containing individual objects can be assembled to reconstruct a full image that can be input into FFB6D or PVN3D.

VII. CONCLUSION

In this paper, we presented CCM Pixel-DA, a novel real-to-sim approach to unsupervised domain adaptation for object pose estimation. Compared with the previous method, the proposed model preserves the structural information, semantic information, and object pose during the transfer. The approach also inherits the advantages of unsupervised domain adaptation methodologies, in that it only requires unpaired pose estimation samples. Thus our method is implemented as a robot grasping system that requires no object pose annotation in the real-world domain. Therefore, the system is low-cost and straightforward to deploy. Moreover, the proposed method decouples domain adaptation and pose estimation, which means that the proposed real-to-sim network can be further applied to the end-to-end learning of other robotic manipulation tasks.

ACKNOWLEDGMENT

This work is sponsored by the Natural Science Foundation of Jiangsu Province (No.BK20201264), Zhejiang Lab (No.2022NB0AB02), and the National Natural Science Foundation of China (No.61573101).

REFERENCES

[1] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, "Blensor: Blender sensor simulation toolbox," in *International Symposium on Visual Computing*. Springer, 2011, pp. 199–208.

[2] B. Planche, Z. Wu, K. Ma, S. Sun, S. Kluckner, O. Lehmann, T. Chen, A. Hutter, S. Zakharov, H. Kosch *et al.*, "Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5 d recognition," in *2017 International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 1–10.

[3] B. Planche and R. V. Singh, "Physics-based differentiable depth sensor simulation," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 387–14 397.

[4] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

[5] S. Zakharov, W. Kehl, and S. Ilic, "Deceptionnet: Network-driven domain randomization," in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 532–541.

[6] S. I. Nikolenko *et al.*, "Synthetic data for deep learning," *arXiv preprint arXiv:1909.11512*, vol. 3, 2019.

[7] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3722–3731.

[8] C. Papaioannidis, V. Mygdalis, and I. Pitas, "Domain-translated 3d object pose estimation," *IEEE Transactions on Image Processing*, vol. 29, pp. 9279–9291, 2020.

[9] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2223–2232.

[10] S. J. Anderson, N. Yamagishi, and V. Karavia, "Attentional processes link perception and action," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 269, no. 1497, pp. 1225–1232, 2002.

[11] M. Tucker and R. Ellis, "The potentiation of grasp types during visual object categorization," *Visual Cognition*, vol. 8, no. 6, pp. 769–800, 2001.

[12] W. Zhao, R. Lun, C. Gordon, A.-B. Fofana, D. D. Espy, M. A. Reinthal, B. Ekelman, G. Goodman, J. Niederriter, C. Luo, and X. Luo, "A privacy-aware kinect-based system for healthcare professionals," in *2016 IEEE International Conference on Electro Information Technology (EIT)*, 2016, pp. 0205–0210.

[13] K. Qian, X. Jing, Y. Duan, B. Zhou, F. Fang, J. Xia, and X. Ma, "Grasp pose detection with affordance-based task constraint learning in single-view point clouds," *Journal of Intelligent & Robotic Systems*, vol. 100, pp. 145–163, 2020.

[14] Y. Liu, L. Zhou, H. Zong, X. Gong, Q. Wu, Q. Liang, and J. Wang, "Regression-based three-dimensional pose estimation for texture-less objects," *IEEE Transactions on Multimedia*, vol. 21, no. 11, pp. 2776–2789, 2019.

[15] 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 *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1530–1538.

[16] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6D object pose estimation by iterative dense fusion," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3338–3347.

[17] S. Iwase, X. Liu, R. Khirodkar, R. Yokota, and K. M. Kitani, "Repose: Fast 6d object pose refinement via deep texture rendering," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 3283–3292.

[18] Y. Xu, K.-Y. Lin, G. Zhang, X. Wang, and H. Li, "Rnnpose: Recurrent 6-dof object pose refinement with robust correspondence field estimation and pose optimization," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 14 860–14 870.

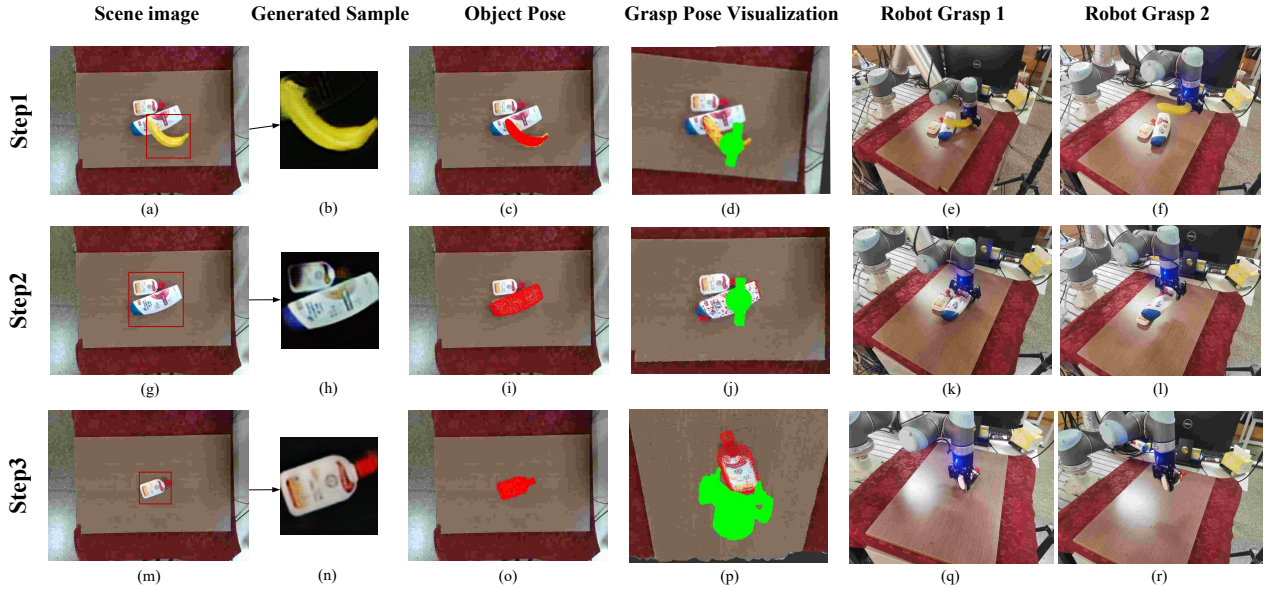

Fig. 12. Online robotic grasping results. Firstly, the RGB-D sensor captured a scene RGB-D image as shown in (a), and the RGB image was transferred to the simulation domain as shown in (b). Then, the object pose was detected as shown in (c) and the grasp pose was shown in (d). (e) and (f) depict the grasping procedure. Step1, Step2, and Step3 depict the sequence of picking up three objects, where the three steps are similar.

[19] Y. Su, M. Saleh, T. Fetzer, J. Rambach, N. Navab, B. Busam, D. Stricker, and F. Tombari, "Zebrapose: Coarse to fine surface encoding for 6dof object pose estimation," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 6728–6738.

[20] Y. He, H. Huang, H. Fan, Q. Chen, and J. Sun, "Ffb6d: A full flow bidirectional fusion network for 6d pose estimation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 3002–3012.

[21] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "PVN3D: A deep point-wise 3D keypoints voting network for 6DoF pose estimation," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 629–11 638.

[22] S. Zakharov, I. Shugurov, and S. Ilic, "Dpod: 6d pose object detector and refiner," in *IEEE/CVF international conference on computer vision*, 2019, pp. 1941–1950.

[23] J.-P. Mercier, C. Mitash, P. Giguere, and A. Boularias, "Learning object localization and 6d pose estimation from simulation and weakly labeled real images," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3500–3506.

[24] 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.

[25] S. Tyree, J. Tremblay, T. To, J. Cheng, T. Mosier, J. Smith, and S. Birchfield, "6-dof pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark," *arXiv:2203.05701v1*, 2022.

[26] H. Cao, L. Dirnberger, D. Bernardini, C. Piazza, and M. Caccamo, "Learning 6d pose estimation from synthetic rgbd images for robotic applications," *arXiv preprint arXiv:2208.14288*, 2022.

[27] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2242–2251.

[28] X. Jing, K. Qian, X. Xu, J. Bai, and B. Zhou, "Domain adversarial transfer for cross-domain and task-constrained grasp pose detection," *Robotics and Autonomous Systems*, p. 103872, 2021.

[29] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12 627–12 637.

[30] X. Liang and E. P. Xing, "Unsupervised real-to-virtual domain unification for end-to-end highway driving," Jun. 6 2019, US Patent App. 16/140,311.

[31] G. Vezzani, U. Pattacini, and L. Natale, "A grasping approach based on superquadric models," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1579–1586.

[32] K. Huebner, "Badgr—a toolbox for box-based approximation, decomposition and grasping," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 367–376, 2012.

[33] G. Vezzani, U. Pattacini, G. Pasquale, and L. Natale, "Improving superquadric modeling and grasping with prior on object shapes," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6875–6882.

[34] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711.

[35] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *arXiv preprint arXiv:1406.2283*, 2014.

[36] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis *et al.*, "Bop: Benchmark for 6D object pose estimation," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 19–34.

[37] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[38] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[39] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *RSS*, 2018.

[40] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4238–4245.

[41] R. Kaskman, S. Zakharov, I. Shugurov, and S. Ilic, "Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects," in *IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[42] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "GraspNet-1Billion: A large-scale benchmark for general object grasping," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 441–11 450.

[43] M. Danielczuk, M. Matl, S. Gupta, A. Li, A. Lee, J. Mahler, and K. Goldberg, "Segmenting unknown 3D objects from real depth images using mask r-cnn trained on synthetic data," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7283–7290.