AUTOMATIC GENERATION OF PHOTOREALISTIC TRAINING DATA FOR DETECTION OF INDUSTRIAL COMPONENTS

Yu-Hui Lee[⋆] *Chu-Chun Chuang*[⋆] *Shang-Hong Lai*[⋆] *Zih-Jian Jhang*[†]

[⋆] National Tsing Hua University, Taiwan
[†] Industrial Technology Research Institute, Taiwan

ABSTRACT

With the prosperous development of deep learning, people pay more attention to the needs of different training data. In this paper, we propose a method to automatically generate realistic training data for industrial components detection. Our method can generate a large scale of various synthetic images associated with the corresponding precise instance segmentation masks through the concept of *domain randomization* and *style transfer*. Besides, we demonstrate that the proposed method is effective to generate images for training a wrench detector. Our method can enhance the performance of the wrench detection task obviously, and it can be easily extended to the detection of different kinds of industrial components. As far as we know, the proposed method is novel and effective for generating training data for training detectors for industrial components.

Index Terms— Synthetic Data, Domain Randomization, Style Transfer, Object Detection

1. INTRODUCTION

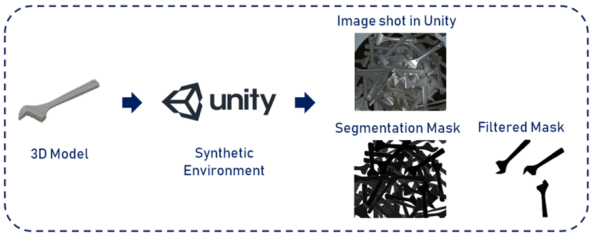
Recent years, deep learning has developed prosperously and made a breakthrough in many computer vision tasks. People are dying to make every corner in the world be more convenience through deep learning, and so is the manufacturing industry. However, to train a high quality deep neural network, one of the most essential factor is to collect a large scale of suitable data. Most of the accessible datasets nowadays are data for objects which are often seen. Since the environment of industrial factory might be quite different from our daily environment, we cannot use common image datasets for specific industrial needs. Besides, there is barely any dataset collected for industrial needs and containing images of various industrial components. Hence, if we want to apply deep learning into industrial tasks, for example, industrial components detection, the best way is to synthesize data ourselves.
With the help of synthetic data, people can acquire data with specific requirements easily and flexibly. Alongside this, synthetic data can also solve the problems that accurate label of some data might be impossible to obtain by hand, or some data might be too sensitive to access.

People has used synthetic data to train a deep neural network for long. A widely used method for generating realistic synthetic data is Generative Adversarial Network(GAN)[1]. For example, Frid-Adar et al.[2] used GAN to synthesize images of liver lesions for training a model of liver lession classification. Wang et al.[3] generated eye images for the gaze estimation task by using GAN as well. Another method for generating realistic data is through rendering images from virtual environment. For instance, SYNTHIA[4] is a synthetic dataset of urban scenes with semantic segmentation mask. Virtual KITTI[5] is also a photo-realistic synthetic video dataset. SYNTHIA and Virtual KITTI are both datasets rendering images from virtual environments.
However, since no matter how realistic the synthetic data looks, there might still be considerable difference between synthetic data and real data. The concept of *Domain Randomization(DR)*[6, 7] was introduced. Domain Randomization is an approach for synthesizing images. Researchers first construct a virtual environment according to the real scene, and then randomly adjust as much parameters in the environment as possible. For example, the light, the texture and the position of each object. Domain Randomization does not focus on rendering the most realistic data, but it tries to generate data as large variations as possible. Once we have simulated images with large varieties, the real images will just be one kind of the rendering conditions. Recently, Tremblay et al. proposed the *Structured Domain Randomization*[8] method. By using both real data and the domain-randomization simulated data, they can improve the performance of object detection in their experiment.
In this paper, we propose a data synthesis method to automatically simulate a large-scale images of industrial components through domain randomization. Meanwhile, our method is able to label the instance segmentation mask for each image generated by our method. Thus, people can significantly reduce the cost of collecting and labelling qualified data from real images. To the best of our knowledge, the proposed method is the first work that can synthesize realistic images and the associated segmentation masks for massive amount of piled industrial components.

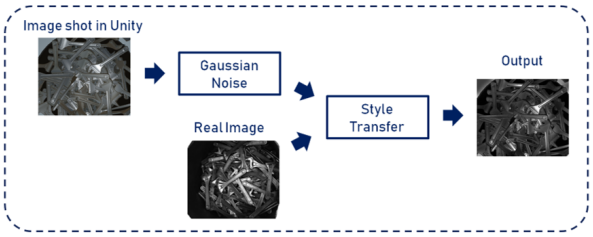
2. PROPOSED METHOD

2.1. System Overview

We proposed a two-step data synthesis method to generate a large scale of realistic images of piled industrial components. The first step is *Image Acquisition* (Figure 1(a)). After collecting the 3D models of a specific industrial component, we build a realistic virtual environment in Unity to automatically generate images of the piled components as well as the associated instance segmentation mask for each photo. However, since it is difficult to fully reconstruct the real scenes, the generated data in virtual environment may still look very different from the real one. Here comes the step of *Photorealism*(Figure 1(b)). We add Gaussian noise and apply style transfer to the generated images to make them be more suitable for object detection tasks. We demonstrate the proposed data synthesis system by applying it to train a wrench detector, and the experimental results show improved wrench detection accuracy by using the proposed data synthesis method.



(a) Pipeline of Image Acquisition



(b) Pipeline of Photorealism

Fig. 1. System Pipeline

2.2. Data Synthesis

2.2.1. *Image Acquisition*

We first construct a virtual environment according to the real scene (Figure 2), included a bucket, point light, 60-80 wrench models, and a camera facing the bucket. Based on *Domain Randomization* [6, 8], we cast about 60-80 wrench models (Figure 3) randomly into the bucket every time an image is synthesized. Each wrench in the bucket has different position and rotation. With the collision detection in the virtual scene, the virtual objects can be precisely placed according to the natural physics phenomenon. Besides, we randomly modified the color and the metallic strength of each wrench model in the scene. After modifying all parameters described above, we can generate a large scale of synthetic images with large variations, thus avoiding the overfitting problem when using these synthesized images.

Fig. 2. Examples of real wrench images

Fig. 3. Different views of the 3D wrench model

2.2.2. *Instance Segmentation Mask Generation*

For the task of object detection, we need precise instance segmentation masks. After setting the realistic material of each object in the virtual scene, we set another material for display the segmentation mask in Unity as well. For each synthetic image of wrenches, we also generate the associated segmentation mask including all of the wrenches in the image (Figure 4, middle). In addition, since our goal is to make a robot pick up wrenches one by one, we produce a filtered mask which preserves only wrenches on the top layer of each pile of wrenches (Figure 4, right). To identify whether a wrench is on the top or not, in the virtual environment, we cast a small number of rays (Figure 5) from different positions of the wrench to the sky. If any of these rays is hit by any other object in the virtual environment, this means the wrench is not on the top layer.

Fig. 4. Instance Segmentation Mask. Left: Synthetic Images, Middle: Full mask, Right: Filtered mask.

2.2.3. *Photorealism*

Since there is no barrier like air when synthesizing images in the virtual 3D environment, the synthetic images would look

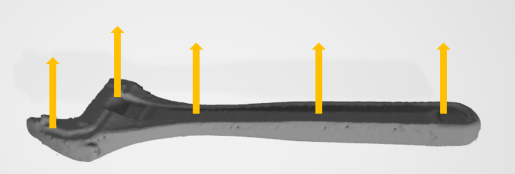


Fig. 5. Illustration of Shooting Rays.

too clear to be realistic. To avoid this problem, we randomly apply Gaussian noise of different levels to the synthetic images.

However, since the synthetic images with Gaussian noise still look quite different from the real images, we then apply style transfer to the synthesized images to make them more realistic.

We first apply the style transfer with CycleGAN[9] since it is able to achieve many kinds of image-to-image style transferring tasks without using any paired data. We train our CycleGAN model with 1,000 synthetic images, and 1,000 real images from SUN[10] dataset. We expect to transfer the synthetic images into the style of real images. However, after the training, we find that the differences between the synthetic images and the general real images are very large. This problem made our results (Figure 6) involve considerable distortion and artifacts. Hence, here we adopt *Fast Photo Style* [11] instead for the style transfer. Fast Photo Style is a method specialized in style transfer, which will be discussed subsequently.

Fig. 6. Results of different style transfer methods. Left: before style transfer, Middle: CycleGAN, Right: FastPhotoStyle

Fast Photo Style can transfer the style of a high-resolution content image into the style of another reference style image through a two-step method, *Stylization* and *Smoothing*. In the *Stylization* step, Fast Photo Style constructs an encoder-decoder generator. The encoder is a VGG-19[12] network(from layers conv1-1 to conv4-1) using ImageNet-pretrained weights[13], and the decoder is the inverse of the encoder trained on Microsoft COCO dataset[14]. With this generator, the style of a content image will be transferred into the style of a reference image temporarily. After that, in the *Smoothing* step, Fast Photo Style would ensure the transferred image looking realistic and remove noticeable artifacts by computing the pixel affinities between the original content image and the temporarily transferred image. Here we set the synthetic image with Gaussian noise as the content image, and randomly pick one out of seven real images of wrenches (Figure 2) as the style image. The results of the *Photorealism* step of the synthetic images after applying Gaussian noise and style transfer are shown in (Figure 7).

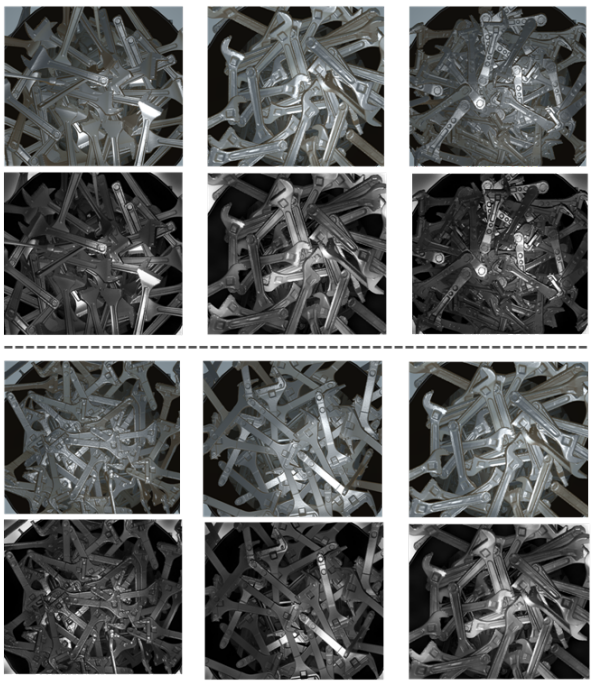


Fig. 7. Photorealism results. Row 1, 3: Before Photorealism, Row 2, 4: After Photorealism.

3. EXPERIMENTAL RESULTS

The proposed method can be separated into two steps, *Image Acquisition* and *Photorealism*. The *Image Acquisition* step is implemented in Unity, on Windows system, and it takes about 1.5 second to generate a pair of the wrench image and the associated segmentation mask on GTX970. The second step, *Photorealism*, is implemented with PyTorch on Linux System. It takes around 2 seconds to transfer the synthetic image to a more realistic image on GTX1080.

Fig. 8. Left: Synthetic Images after Photorealism, Middle: Full mask, Right: Filtered mask.

To evaluate how much our image synthesis method improves the object detection task, we use MASK R-CNN[15] as our wrench detector, and set the weight of mask loss 10 times bigger than the other loss. Here we implement 4 kinds of experiments, training wrench detectors on *only real data*, *only our synthetic data*, *both real and synthetic data*, and *both real and synthetic data (but no style transfer)*.

For real data, the dataset consists of 220 real-captured images with manually-labeled segmentation mask (Figure 9). These segmentation masks only locate wrenches which are not occluded by other objects (on the top of each wrench pile). We randomly pick 191 images in the dataset as training data, and simply flip these images to obtain twice the amount of real images for training. For the remaining 29 images in the dataset, we use them as testing images. Besides, we generate 1,000 images through our method as synthetic training data.

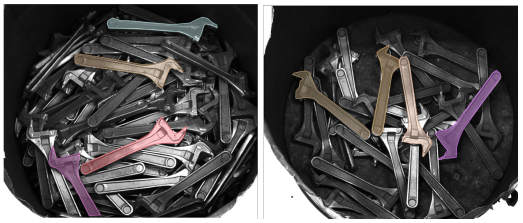


Fig. 9. Illustrations for the ground truth segmentation of real data. Instance segmentation is shown in colored mask.

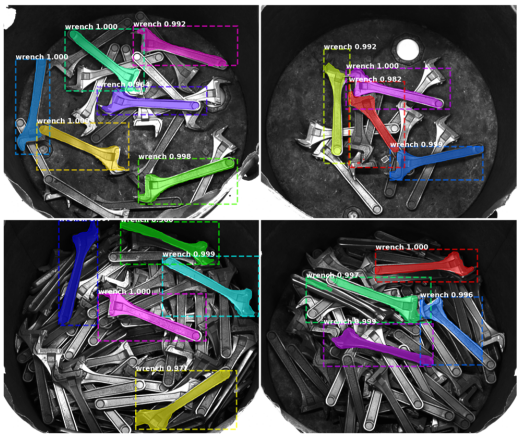


Fig. 10. Qualitative results on real data. Wrench detector is trained on both real and our synthetic data. Predicted results are shown in colored masks.

The quantitative result is shown in Table 1 and Table 2. The testing set contains 29 real images of wrench pile. Table 1 shows the Average Precision (AP) of detectors evaluated at 0.5 IoU overlap. When training a detector with either real or synthetic data, the difference in the AP for mask segmentation is not much as shown in Table 1. However, after extending the training data with real data and our synthetic images, the mask AP is increased 20 points, achieving to **78.2**, and the bounding box AP is increased 10 points, achieving to **82.0**. This indicates that the proposed synthetic data generation method significantly enhances the accuracy of the wrench detection task. Even training detector with both real and synthetic data without style transfer, mask AP and bounding box AP are improved. Some examples of the detection results are depicted in Figure 10. The wrench detector trained on both real and our synthetic images can generate high-quality instance segmentation masks and bounding boxes for the detected wrenches in the wrench pile images. Apart from this, as shown in Table 2, the more synthetic images we use for training, the better the wrench detector can perform.

Table 1. Average Precision @0.5 IoU of using different training data for the wrench detection task. Real data(**R**): 392 images, Synthetic data(**S**): 1,000 images.

Training Data	mask AP	bbox AP
Only R	57.1	72.0
Only S (w/o style transfer)	54.4	60.4
Only S (w/ style transfer)	59.0	64.7
R and S (w/o style transfer)	73.0	76.9
R and S (w/ style transfer)	**78.2**	**82.0**

Table 2. Average Precision @0.5 IoU of using both 392 **real images** and **different numbers of synthetic images(with style transfer)** to train a wrench detector.

Training Data	mask AP	bbox AP
500 Synthetic images	71.4	74.6
1,000 Synthetic images	78.2	82.0
2,000 Synthetic images	79.8	80.2
4,000 Synthetic images	**82.0**	**82.9**

4. CONCLUSIONS

In this paper, we proposed a method to automatically generate realistic training data for industrial components detection. The proposed method is the first work that can synthesize realistic images and associated segmentation masks for a massive amount of piled industrial components. We do not only focus on simulating real data, but also care about the diversity of synthetic data. Based on domain randomization, we generate a large number of diverse synthetic images and their precise instance segmentation mask. Besides, we reduce the gap of synthetic images and the real images through the style transfer technique. Our experiments show the proposed method dramatically improve the accuracy of wrench detection, and we believe that this method can be easily extended to the detection task of other industrial components.

5. REFERENCES

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[2] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan, "Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification," *arXiv preprint arXiv:1803.01229*, 2018.

[3] Kang Wang, Rui Zhao, and Qiang Ji, "A hierarchical generative model for eye image synthesis and eye gaze estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 440–448.

[4] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

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

[6] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 23–30.

[7] Joshua Tobin, Lukas Biewald, Rocky Duan, Marcin Andrychowicz, Ankur Handa, Vikash Kumar, Bob McGrew, Jonas Schneider, Peter Welinder, Wojciech Zaremba, et al., "Domain randomization and generative models for robotic grasping," *arXiv preprint arXiv:1710.06425*, 2017.

[8] Aayush Prakash, Shaad Boochoon, Mark Brophy, David Acuna, Eric Cameracci, Gavriel State, Omer Shapira, and Stan Birchfield, "Structured domain randomization: Bridging the reality gap by context-aware synthetic data," *arXiv preprint arXiv:1810.10093*, 2018.

[9] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *arXiv preprint*, 2017.

[10] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE, 2010, pp. 3485–3492.

[11] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz, "A closed-form solution to photorealistic image stylization," *arXiv preprint arXiv:1802.06474*, 2018.

[12] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 2009, pp. 248–255.

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask r-cnn," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2980–2988.