Synthetic Training Data Generation for Deep Learning-Based Billet Detection in Rolling Mills

Maria LUSCHKOVA [1], Christian SCHORR [1,2], Tim DAHMEN [1]
[1] German Research Center for Artificial Intelligence, Saarbrücken, Germany
[2] University of Applied Sciences Kaiserslautern, Zweibrücken, Germany
Contact e-mail: christian.schorr@dfki.de

Abstract. AI-powered quality assurance solutions are gaining momentum in the steel industry under the Industry 4.0 paradigm. In rolling mills, knowing the real-time location of billets, i.e. fast moving bars of hot steel, is important in order to guarantee a safe process and defect-free end products. To achieve this aim, we present a deep learning-based detection of these billets in rolling mills using synthetically generated training data. A core practical challenge for many deep learning projects is the limited availability of appropriate, annotated training data. We propose a method for simulating images employing a partial digital twin of the rolling process. Partial models governing the shape and location of the billets, the layout of the rolling mill floor, the camera settings, and the lighting situation changing over time are combined into a scenario model. Choosing different parametrizations of this scenario model facilitates synthesizing a broad range of images for training. The resulting deep learning model is utilised to detect billets in real-world images from an actual rolling mill. We describe the creation of the partial models using aerial photogrammetry, expert knowledge, and 3D modelling, as well as the choice of the deep learning model. An evaluation of the model's performance on real-word images shows the applicability of our synthetic training data approach.

1. Introduction

As a result of automation and demands for increased productivity, reduced human intervention, and improved workplace safety, today's industries are placing considerable emphasis on process inspection and quality control. In the context of steel production, a challenging problem is the tracking of individual billets in the rolling mill, providing linkage between sensor data on individual billets collected before and after the non-continuous blooming train. Before entering a heating furnace, each billet is tagged by an imprinted billet ID, which is destroyed during the rolling process, making it impossible to identify it at the end of the rolling process. Due to harsh production conditions in the steel mill, technologies such as RFID sensors are also unsuitable for the tracking task. A computer vision-based tracking system provides an alternative, as cameras can be placed at a distance to the field of observation, protecting them from damage. The available data for the tracking system consists of video streams from three Full HD cameras placed at the entrance, in the middle, and at the exit of the blooming train. A schematic illustration of the site as well as captured images from the three cameras are depicted in Figure 1 and Figure 2.

Due to the complexity of the task of tracking billets over multiple camera views, in this paper we focus on the initial tracking step: detecting individual billets in the scenery. State-of-the-art object detection methods mostly use deep learning technology. The strength

of the approach lies in the ability to learn rich representations as well as to automatically extract relevant features from the training data. However, one of the main obstacles using deep supervised learning for computer vision-based inspection is the shortage of annotated training data. Training data captured from physical processes does have data distributions that follow the physics of the underlying process. For stablished production processes, accidents and defects happen rarely if at all. Consequently, the most relevant cases are drastically underrepresented in training data captured in the real world. An encouraging solution is to generate labeled training data using an image data simulator [1]. High-fidelity simulations enable training and testing deep learning algorithms more effectively, leading to more robust and adaptive networks. Models are able to gain considerably more experience in the photorealistic virtual world than in the real environment. It is not only possible to simulate rare events that pose challenging situations, e.g. appearance of abnormalities in industrial processes, but also to generate broadly distributed variations in a data set, enabling the model to better generalize in cases of unseen data.

In this paper, we evaluate the impact of utilising simulated data for billet detection in a rolling mill. After reviewing data simulation strategies, we present a simulation pipeline for creating training data of extensive variance. We begin by demonstrating the transfer of a real blooming train to its digital counterpart and populating it with billet models. Then we explain how the rendering process is automated. At the end we choose a deep learning model to check the quality of the rendered data.

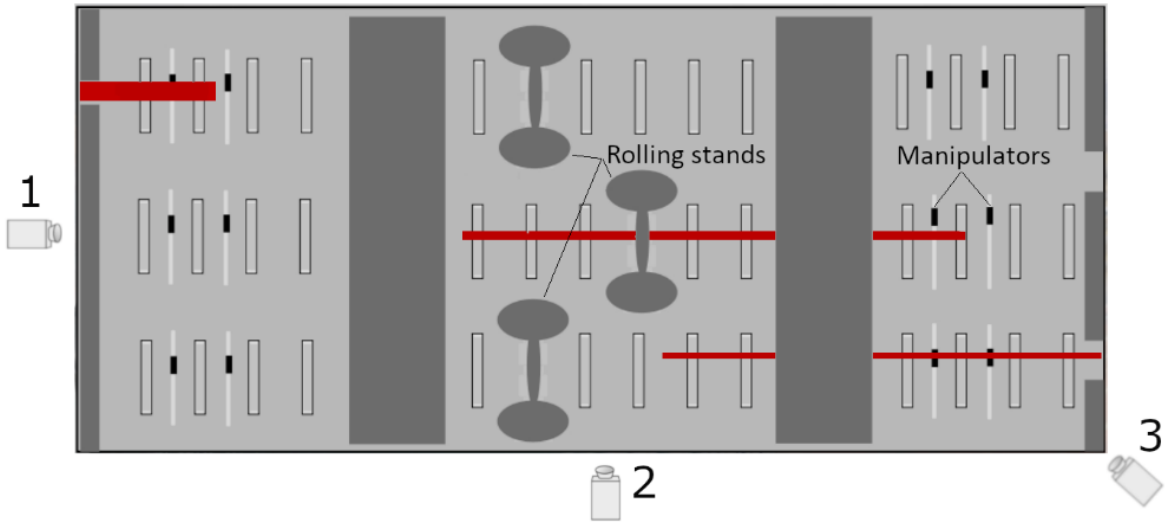

Figure 1 Schematic illustration of a blooming train with billets and three cameras. Image courtesy of Saarstahl AG.

Figure 2 Captured images from blooming train cameras.

2. Related Work and State of the Art

The use of simulated data has been extensively explored in various domains of computer vision [2], [3]. A systematic approach to the idea was proposed in [4] and is mostly followed in this study. Due to the data-intensive nature of deep convolutional neural networks, simulations are frequently employed for tasks where human annotation of the data is either difficult or expensive. Simulated data not only makes data labeling more cost effective, but also permits learning theoretical cases for which no real data exists.

One problem when working with simulated image data is the domain gap issue [5], where models trained on simulated data cannot generalize well to real data. This is due to difficulties to include all factors in image simulators.

There are three techniques to help surpass this obstacle: improved photorealism, synthetic domain randomisation, and domain adaptation. Improved photorealism is essential for training strong detectors. To produce computer-generated images with a high level of visual realism, one must focus on modeling a scene with great attention to geometry, textures, lightning conditions, as well as accurate simulation of camera lenses and their image formation. [6] and [7] demonstrated benefits using photorealistic rendering for training region-proposal-based object detectors.

Domain randomization is one of the most promising techniques to make transfer learning from the synthetic image domain to the real image domain work. The basic idea of domain randomization is to generate synthetic data diverse enough to train robust models that operate efficiently on real data. In computer vision, there are various ways to randomize synthetic images: While domain randomization intentionally avoids photorealism and opts for a variety in synthetic images [5], [1], structured domain randomisation procedurally generates synthetic random images preserving the structures and context of the problem domain [8], [9]. Instead of employing 'blind' domain randomization, [10] and [11] explore how to choose the best possible parameters for guided domain randomization.

Domain adaptation strategies endeavor to make a model trained in a synthetic domain perform effectively in a real domain. These strategies can be divided into two main groups: The first group works with the synthetic data itself, trying to 'refine' it to the real domain by making the 'fake' data more realistic [12], [13]. A few studies, however, have demonstrated that synthetic images may work better if they look less realistic, resulting in better model generalization. The methods from the second group operate directly at the feature and model level [14], [15]. These methods perform a synthetic-to-real domain adaptation, although they do not necessarily result in more realistic synthetic data.

3. Methods

3.1 Synthetic Training Image Generation

By creating virtual environments, we are able to generate training data in a controllable way, avoiding collecting and annotating real world data [16], [17], [18]. The major investment in labour time occurs at the beginning of the configuration process, when all visual assets of modelled scenery must be configured. Once the simulated environment is set, a render engine is used to generate labeled data with randomized scenario parameters. Parameter randomization is essential for introducing variations into the generated training data. After a certain amount of rendered images, one validates the already rendered data using a deep learning model and real data. In case the rendered data exhibits unacceptable domain gap, it might become necessary to adapt the models.

3.2 Environment Generation

In this study, we model the bare environment based on the real blooming train setting, to represent static obstacles in the scenery such as rollers and control booths. A 3D model of the blooming train is computed by close-range photogrammetry [19]. High-resolution photos of the rolling street were captured using a Canon EOS 5D at 12.8 Megapixel resolution and converted to a 3D reconstruction using the commercial software Agisoft Metashape. This 3D model replicates the real environment in roughly correct dimensional proportions (Figure 3).

After the model of the blooming train was cleared of unnecessary objects and prepared for further use, three virtual cameras were placed in it at locations predetermined by the position of the actual cameras. To make the generated dataset robust to slight changes in camera position, we varied the camera rotation and coordinates within a range of a few degrees and centimetres. We also applied a wide-angle lens distortion to reflect real images.

The 3D setting was composed with original camera shots taken during production downtime. Hourly photos of the idling rolling mill captured over two weeks convey a wide range of changes in lightning conditions. Additionally, we considered using 3D renderings of the photogrammetry mesh as background but discarded the approach as the texturing and postprocessing of the photogrammetry mesh constituted in infeasible effort. These difficulties are a result of the extreme amount of geometric detail in the milling plant and are likely surpassable for simpler scenarios.

3.3 Billet Generation

The billets detected in this study are thin elongated hot steel bars whose thickness decreases and length increases during the rolling process. Billets are not always rigidly straight, they can be bent and deformed as they are moved or rolled. If a billet is deformed only slightly, it can be straightened in the following rolling rounds. In case of severe deformation, the billet is removed from the blooming train and recycled. The color and luminance of billets change as they cool down. Furthermore, the billet's apparent color is liable to change as a result of daytime-dependent lighting changes and resulting automatic adjustments in-camera, as well as optically inherent defects, such as chromatic aberrations.

An important part of modelling billets is to decide which shape representation to use. We assume that the billet shape has the form of circle or square extruded along a curved profile defined by a B-spline. In our virtual environment, the simulated material of a billet consists of two components. The first component, the material color channel, is the color of the billet itself. In this study, we are focusing on detecting only hot billets. Exiting directly out of the furnace and during the rolling process, such hot billets are recorded by the camera in an oversaturated white color. The second component, the luminance channel, is affected by extraneous factors. As noted previously, while the physical luminance of the actual hot billets remains unchanged, the perceived luminance level and tint in-camera depend on the surrounding conditions like day and night illumination and individual camera properties and settings. Therefore, we randomize the shader's luminance tint between pink and purple in the hue channel of the HSV color space.

For each scenario, three to seven billets are instantiated and randomly positioned on the floor of the mill model. To ensure that there is no unrealistic overlap between the billets and the rest of the environment, they are placed in a predefined masked area of the blooming train. The density of billets on the floor is also randomized so that the billets can lie very close to each other, as in real-world scenarios, Figure 4.

3.4 Rendering Pipeline Automation

Training data was generated by randomizing environmental and billet parameters for every frame. The parameters characterizing the environment are the number of billets in the scenery, the camera position, the camera rotation, and the background image. Billets in turn are described by their length, thickness, degree of deformation, color, luminance, and position in the mill. JSON files with extensible schema are employed to programmatically write and read randomized scenario parameters. Based on this set of input parameters, the image simulator compiles a 3D scene and renders it in three camera views.

We additionally considered rendering entire animations. The approach was discarded because from the point of view of training set composition, having several very similar frames is inferior to a more random data distribution.

The process of populating the dataset with new images happens on-demand. The main limiting factor of adding new images is the time needed to render each image. On average, one minute is needed to render an image of size 1920x1080 with a corresponding ground-truth billet instance mask on Nvidia Tesla V100S with 32 GB RAM. Samples of rendered images are presented in Figure 5. This kind of synthetic images with corresponding masks is employed to train an object detection network.

3.5 Billet Detection with Deep Learning

In this study, we evaluate synthetic training data for billet detection in real-world videos. As a billet detector we chose a well-developed Faster R-CNN architecture consisting of two networks: a region proposal network to generate region proposals and a network using these proposals to detect objects. The method requires image annotations to be dictionaries with bounding box coordinates and object labels. These annotations are automatically generated from rendered billet instance masks. We discarded such billet instances from annotating if the area of their bounding boxes was less then 100 pixels, since Faster-RCNN is known to encounter difficulties detecting very small objects [20].

Figure 3 A 3D model of the Saarstahl blooming train, reconstructed photogrammetrically.

Figure 4 Left: A 3D model of the Saarstahl blooming train with inserted 3D billet models. Middle: A 3D setting composed with original camera footage. Right: Billet instance segmentation masks with false color IDs.

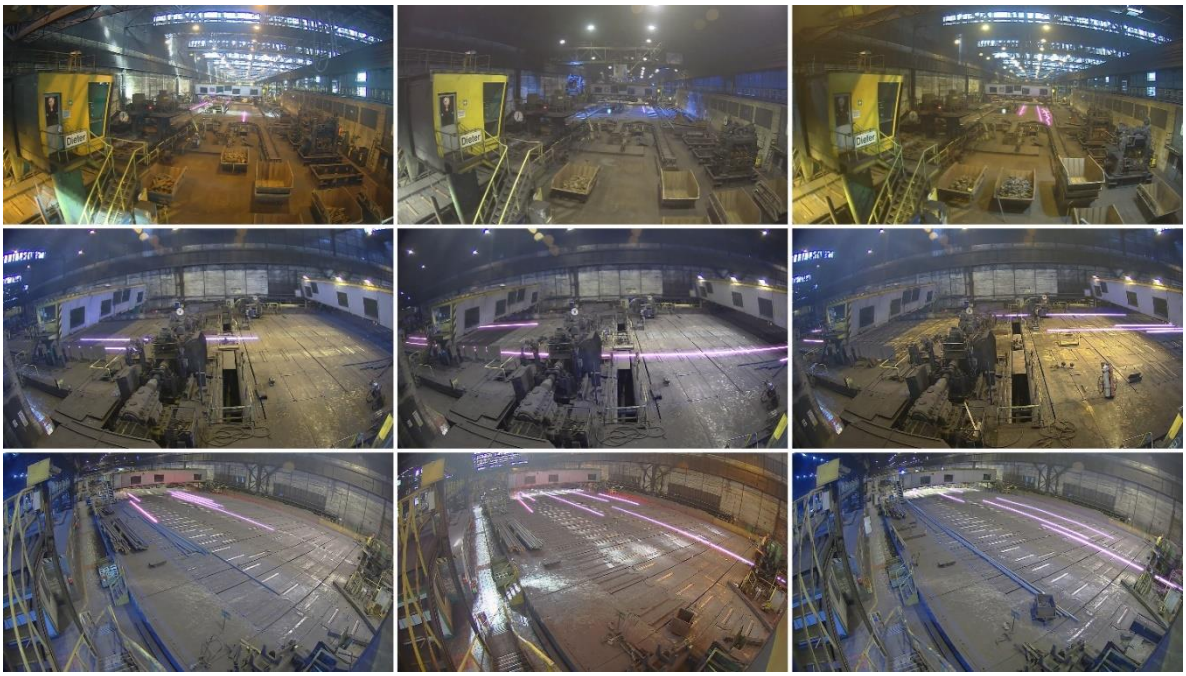

Figure 5 Examples of synthetic images generated automatically with the proposed approach.

4. Experiments

Using the approach discussed above, we generated a randomized synthetic dataset containing approximately 30,000 images with instance segmentation masks for each camera view. We trained deep learning networks using only the generated data. As computing infrastructure, we utilized a server with Ubuntu 18.04 with one Nvidia Tesla V100S with 32 GB VRAM. We used the PyTorch implementation of the Faster R-CNN [21] architecture with the ResNet50 [22] backbone pre-trained on the COCO dataset [23]. To adapt the pretrained network to our target domain, we performed transfer learning by replacing the pretrained network head by a layer with two classes: background class and billet class. The SGD optimizer is used with a learning rate of 0.005, momentum of 0.9, and weight decay of 0.0005, with minibatch size of 4 images, and training run for a maximum of 20 epochs. Input image size is set to 1920x1080 pixels in all experiments. The synthetic dataset is split into 80% for training and 20% for validation. 323 labeled real images were used as a test set for the evaluation of the synthetic data and model quality.

Experiments were performed on training datasets with different numbers of simulated images (with 1.000, 5.000, 10.000, 30.000). Each of the four models is subsequently evaluated on the real data. Figure 6 shows the bounding box detection average precision (AP) and losses over 20 epochs on the synthetic validation dataset. Table 1 lists the IoU metric for billet bounding box detection on real test images. Detection AP, losses, and IoU are shown for the second camera view. In all four categories, training on 30.000 synthetic images leads to the best result. Figure 6 indicates that increasing the synthetic dataset from 1.000 to 5.000 images improves training by a large margin, while increasing the dataset from 10.000 to 30.000 images improves training only slightly. This stagnation of improvements can be explained by the fact that synthetic validation sets do not differ a lot between 10.000 and 30.000 datasets. However, Table 1 shows clear improvement in the training between 10.000 and 30.000 datasets on real test images.

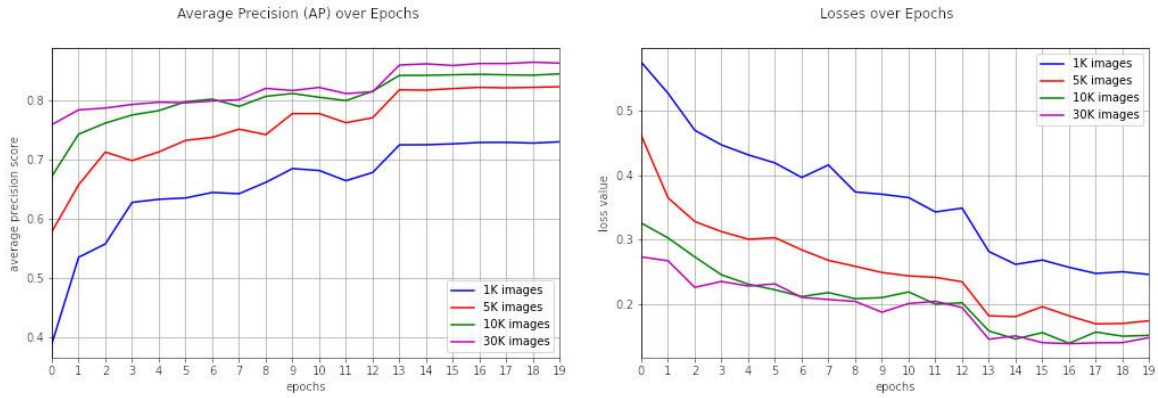

Figure 6 Bounding box detection average precision and losses on the synthetic validation set over training epochs.

Table 1 Performance results of IoU of each model on the real dataset after 20 epochs.

Number of images in synthetic training dataset	IoU
1.000	0.676
5.000	0.704
10.000	0.701
30.000	0.744

5. Evaluation

Once trained on simulated images, Faster-RCNN is presented with real-world images from the billet rolling process and tasked to detect billets in the scenery. Example detection results for three camera views are provided in Figure 7. It is obvious that Faster-RCNN is effective for finding the location of billets in images. Even if simulations are hardly realistic for a human observer, their learnable features are still good enough to be transferred to the real image domain. The high detection threshold of 0.9 confirms this statement.

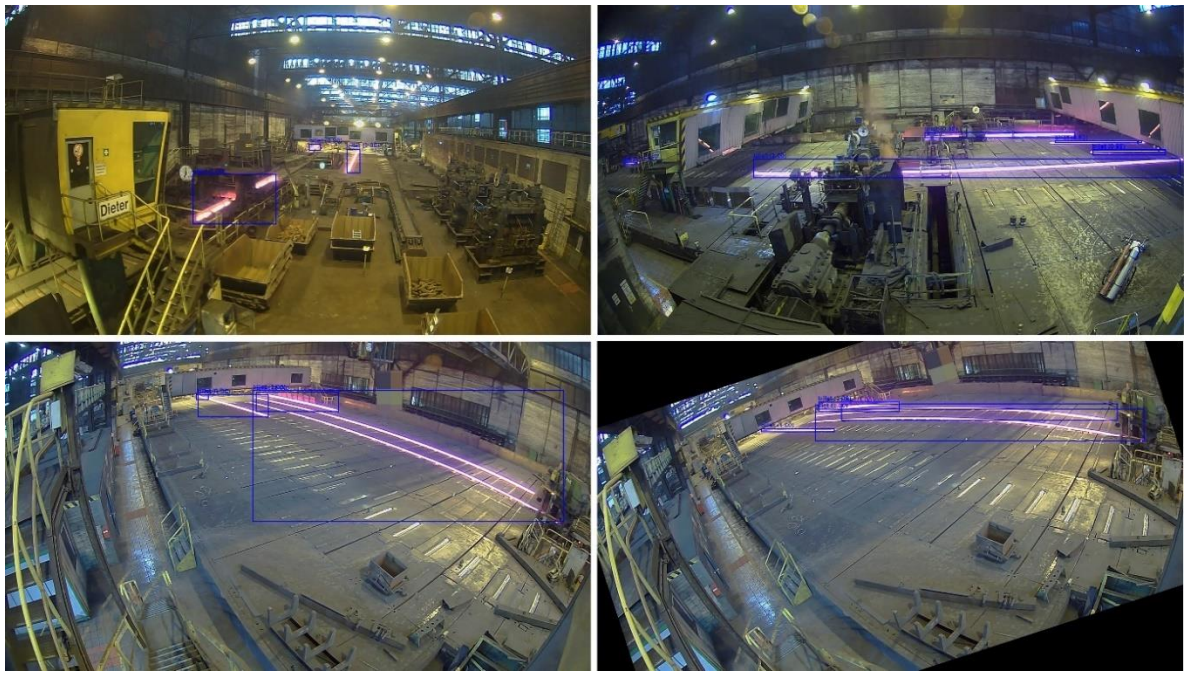

Figure 7: Billet detection in real images

In camera 3, however, we observe misdetections in case the billets lie in close proximity to each other. Due to the horizontal anchors of Faster-RCNN and extremely long and thin shape of the steel bars, the billets receive one bounding box. The problem with rotated objects can be solved by using oriented proposals [24], [25]. In our use case, there is only one angle of billet rotation in the scenery, so we managed to escape the predicament by re-training Faster-RCNN with rotated simulated images, see Figure 7 bottom right.

6. Conclusion

Here, we introduced a pipeline for synthesizing images of a rolling mill and proved that deep learning models trained solely on synthetic data do learn distinctive features for billet detection. Presenting the simulation pipeline was the primary emphasis of our paper, while tuning the detection model was not given high priority.

Together with our previous work [26], [27], the demonstrated pipeline advances us another step closer to creating a versatile toolkit with various software components to generate synthetic training data for industrial processes where real world data is not available or annotation constitutes an infeasible effort. A disadvantage of the approach is that each new problem requires modeling a virtual scenario environment before the data can be rendered, which might take a significant amount of time, depending on the complexity of a scenery. This motivates additional computer graphics research on the automatic or semi-automatic generation of parametric generative models.

While our current work has focused on the task of billet detection, a future step should be in the direction of billet tracking, which may raise new problems related to the availability of training data. If so, we believe that the presented methodology can be transferred to the task of synthesizing training data for billet tracking.

Acknowledgements

This research was funded by the European Union's Horizon 2020 research and innovation programme under grant number 870130 (the COGNITWIN project). The authors thank Saarstahl and Stahl-Holding-Saar (SHS) for their technical support and videos from the blooming train.

References

[1] 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," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops,* pp. 969-977, 2018.

[2] A. Tsirikoglou, G. Eilertsen and J. Unger, "A survey of image synthesis methods for visual machine learning," *Computer Graphics Forum, vol. 39 (6),* pp. 426-451, 2020.

[3] S. I. Nikolenko, "Synthetic data for deep learning," *arXiv:1909.11512,* 2019.

[4] T. Dahmen, P. Trampert, F. Boughorbel, J. Sprenger, M. Klusch, K. Fischer, C. Kübel and P. Slusallek, "Digital reality: A model-based approach to supervised learning from synthetic data," *AI Perspectives 1,* pp. 1-12, 2019.

[5] 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," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2017),* pp. 23-30, 2017.

[6] T. Hodan, V. Vineet, R. Gal, E. Shalev, J. Hanzelka, T. Connell, P. Urbina, S. N. Sinha and B. Guenter, "Photorealistic image synthesis for object instance detection," *2019 IEEE International Conference on Image Processing (ICIP),* pp. 60-70, 2019.

[7] D. Dwibedi, I. Misra and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," *Proceedings of the IEEE International Conference on Computer Vision,* pp. 1301-1310, 2017.

[8] A. Prakash, S. Boochoon, M. Brophy, D. Acuna, E. Cameracci, G. State, O. Shapira and S. Birchfield, "Structured domain randomization: bridging the reality gap by context-aware synthetic data," *2019 International Conference on Robotics and Automation (ICRA),* pp. 7249-7255, 2019.

[9] L. Eversberg and J. Lambrecht, "Generating images with physics-based rendering for an industrial object detection task: Realism versus domain randomization," *Sensors vol. 21(23),* 2021.

[10] Q. Vuong, S. Vikram, H. Su, S. Gao and H. I. Christensen, "How to pick the domain randomization parameters for sim-to-real transfer of reinforcement learning policies?," *ArXiv, abs/1903.11774,* 2019.

[11] S. Zakharov, W. Kehl and S. Ilic, "DeceptionNet: network-driven domain randomization," *2019 IEEE/CVF International Conference on Computer Vision (ICCV),* pp. 532-541, 2019.

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

[13] J. Katrolia, L. Krämer, J. Rambach, B. Mirbach and D. Stricker, "An adversarial training based framework for depth domain adaptation," *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, vol. 4: VISAPP,* pp. 353-361, 2021.

[14] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. March and V. Lempitsky, "Domain-adversarial training of neural networks," *Journal of Machine Learning Research,* pp. 1-35, 2016, vol.17 (59).

[15] G. French, M. Mackiewicz and M. Fisher, "Self-ensembling for visual domain adaptation," *International Conference on Learning Representations (ICLR),* 2018.

[16] S. Borkman, A. Crespi, S. Dhakad, S. Ganguly, J. Hogins, Y.-C. Jhang, M. Kamalzadeh, B. Li, S. Leal, P. Parisi, C. Romero, W. Smith, A. Thaman and S. Warren, "Unity perception: Generate synthetic data for computer vision," *arXiv preprint arXiv:2107.04259,* 2021.

[17] S. E. Ebadi, Y.-C. Jhang , A. Zook , S. Dhakad , A. Crespi , P. Parisi , S. Borkman , J. Hogins and S. Ganguly, "PeopleSansPeople: A synthetic data generator for human-centric computer vision," *eprint arXiv:2112.09290,* 2021.

[18] A. Kar, A. Prakash, M.-Y. Liu, E. Cameracci, J. Yuan, M. Rusiniak, D. Acuna, A. Torralba and S. Fidler, "Meta-Sim: learning to generate synthetic datasets," *2019 IEEE/CVF International Conference on Computer Vision (ICCV),* pp. 4550-4559, 2019.

[19] F. Remondino, A. Guarnieri and A. Vettore, "3D modeling of close-range objects: Photogrammetry or laser scanning?," *Proceedings of SPIE-IS&T Electronic Imaging: Videometrics VIII 5665,* pp. 216-225, 2004.

[20] C. Eggert, S. Brehm, A. Winschel, . D. Zecha and . R. Lienhart, "A closer look: Small object detection in faster R-CNN," *2017 IEEE International Conference on Multimedia and Expo (ICME),* pp. 421-426, 2017.

[21] S. Ren, K. He, R. B. Girshick and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal," *Advances in Neural Information Processing Systems (NIPS),* pp. 91-99, 2015.

[22] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 770-778, 2016.

[23] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick and P. Dollár, "Microsoft COCO: Common objects in context," *ECCV,* 2014.

[24] X. Xie, G. Cheng, J. Wang, X. Yao and J. Han, "Oriented R-CNN for object detection," *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV),* pp. 3520-3529, 2021.

[25] . L. Zhang, H. Wang, L. Wang, C. Pan, Q. Liu and X. Wang, "Constraint loss for rotated object detection in remote sensing images," *Remote Sensing, vol. 13,* p. 4291, 2021.

[26] P. Gutierrez, M. Luschkova, A. Cordier, M. Shukor, M. Schappert and T. Dahmen, "Synthetic training data generation for deep learning based quality inspection," *International Conference on Quality Control by Artificial Vision,* 2021.

[27] P. Trampert, D. Rubinstein, F. Boughorbel, C. Schlinkmann, M. Luschkova, P. Slusallek, T. Dahmen and S. Sandfeld, “Deep neural networks for analysis of microscopy images—synthetic data generation and adaptive sampling,” *Crystals, vol. 11(3),* 2021.

[28] F. Poucin, A. Kraus and . M. Simon, “Boosting instance segmentation with synthetic data: A study to overcome the limits of real world data sets,” *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW),* pp. 945-953, 2021.