Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 100 (2021) 714–719

www.elsevier.com/locate/procedia

31st CIRP Design Conference 2021 (CIRP Design 2021)

# Leveraging synthetic data from CAD models for training object detection models – a VR industry application case

Sampsa Kohtala*, Martin Steinert

*Norwegian University of Science and Technology, Department of Mechanical and Industrial Engineering, Richard Birkelands vei 2B, 7491 Trondheim, Norway*
* Corresponding author. Tel.: +47-413-58-744; *E-mail address:* sampsa.kohtala@ntnu.no

**Abstract**

In this paper we evaluate the applicability of using synthetic data, based on computer aided design models, to automatically detect objects in the real world. The aim is to enable scalable deep learning-based object detection to track and identify physical objects using a single low-cost camera. The approach is demonstrated and evaluated through a case-study involving a physical scale-model of an industrial plant connected to a virtual environment, aimed at facilitating multidisciplinary collaboration and immersive visualization. The digital models are simulated using domain randomization, and subsequently used to train object detection models. The results show the methods' ability to generalize to real data, with accuracies up to 87%, demonstrating the scalability of the approach. Potential applications in industry are discussed based on these results.



*Keywords:* Object Detection; Synthetic Data; Virtual Environment

## 1. Introduction

The use of computer aided design (CAD) tools supports the creation, development and realization of products and is widely used in product development and production industries. CAD models are traditionally used for generating and visualizing concepts, decision making, analyzing and optimizing designs, and refining models prior to manufacturing. In this paper we evaluate and demonstrate the potential of utilizing existing CAD models beyond their traditional use-cases in the context of Industry 4.0 and cyber-physical systems, by connecting the physical product to its existing digital model.

A method for automatically recognizing and tracking physical objects based only on their digital models is presented through a case study. The case involves a physical scale model connected to a virtual environment (VE), aiming to provide an intuitive platform to facilitate multidisciplinary collaboration and immersive visualization [1]. After producing the digital models through e.g., 3D-printing, the objects are automatically tracked and connected to a VE, allowing users to interact with the models physically to test different configurations, and then view them in a real-scale environment through virtual reality (VR).

The tracking and recognition of objects in the real world are realized using a single low-cost webcam, by training deep learning-based object detection models using synthetic data. Synthetic data are generated by simulating the CAD models in Unity using domain randomization (DR). The synthetically trained object detection models should also be reliable in various environments, i.e., when the appearance of CAD models and their surroundings are not known prior to production or use-cases. By comparing different approaches for generating the data, we demonstrate the methods' ability to generalize for detecting objects in the real world, even when the appearance of the environment or products are unknown in advance.

The presented case study is currently intended for projects involving configuration problems, e.g., the planning of construction sites, shipyards, or real-estate, although the method for automatically training object detection models is relevant in other areas such as robotic vision, production


10.1016/j.procir.2021.05.092

monitoring and digital twinning of physical assets. Thus, our aim is to develop a generic method for automatically training object detection models based on digital CAD models. The method is especially useful for small and medium-sized enterprises in the context of digital transformation, by alleviating the labour intensive and costly process associated with manually producing training data for object detection.

A short introduction to related systems is presented, followed by previous work on using synthetic data for training object detection models. The remaining sections focuses on the approach of generating synthetic data before training and evaluating the models in the results section. A discussion follows on the results with potential applications and limitations.

## 2. Related work

### *2.1. Combining physical and virtual environments*

In product development and design, digital tools are used to generate and visualize concepts, while 3D-printing can produce a tangible experience based on its digital model, thus supporting further testing and decision making. To combine the benefits of both the physical and digital prototyping mediums, Min, et al. [2] used object tracking and hand gesture recognition together with an AR headset to superimpose virtual elements on the physical model, including both visualization and interaction possibilities. By combining physical and digital elements they were able to increase the fidelity of the prototype, supporting more testing and decision making during the early design phase, without having to invest allot of resources in producing a real functional product. The object tracking method was implemented using the existing Vuforia software development kit, which only supports a maximum of two objects to be tracked simultaneously.

Similar to our case study, Arrighi and Mougenot [3] developed a tangible user interface for manipulating a VE. Their goal of supporting user participation in the early design process was achieved with an immersive VR application supported by a simple interface. The interface consisted of 3D-printed objects with markers, which were tracked by a camera and then represented in the VE in real time, allowing non-expert users to interact naturally with virtual elements. The tracking method used requires custom markers to detect each individual object, thus making the system less scalable and more dependent on the developer.

The synthetic prototype environment by Damgrave and Lutters [4] allows multiple users to collaborate using a tangible interface connected with their virtual elements. With focus on Industry 4.0 production environments, their synthetic prototype environment consists of a scaled version of a real or potential environment, providing an overview with tangible interaction possibilities for multiple stakeholders, and supports easy and effective visualization and decision making through the digital tools.

A cyber-physical collaboration tool was introduced in [1], shown in Fig. 1., where the physical model (c) acts as the interactive interface for the VE (d).

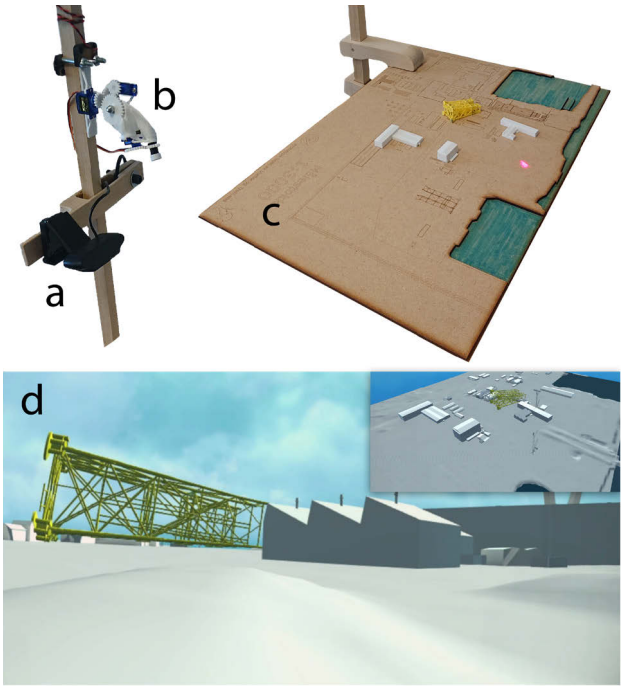


Fig. 1. Components of the cyber-physical collaboration tool, consisting of (a) a webcam for tracking, (b) laser pointer for showing the location and viewing angle of the VR-user, (c) a physical (2D) scale model with 3D printed objects, and (d) the corresponding real-scaled virtual environment shown from the point of view of the VR-user.

The physical setup aims to promote engagement and collaboration, while the VR application provides possibilities for detailed inspection, quality assurance and immersion etc., creating a synergic effect between the two components during multidisciplinary collaboration. A camera-based tracking method was implemented, utilizing deep learning-based object detection, to connect the physical objects to the VE. This approach required the tedious process of manually capturing training data to train the object detection model.

To make the systems and related methods scalable and easy to implement, a simple way to connect the physical models to their virtual representations must be realized. Similarly to [4], there should be an information backbone that can connect every object, tool and software used. Since we can assume that a digital object is created first, this information should be utilized to autonomously track the physical objects, e.g., before 3D-printing is initiated, thus being automatically prepared for the physical and virtual environments without any extra steps. This will also allow (non-expert) users to seamlessly add new content to the system, by only registering a new digital model to be 3D-printed and used in the physical and virtual environments.

### *2.2. Training object detection models with synthetic data*

Traditional deep learning-based object detection requires many labeled training samples captured from the real world to perform well, which can be cumbersome, time-consuming, and thus expensive to create. As stated by [5] in their application of detecting parts for robotic operations, creating datasets for many parts is tedious and an automated process is needed. Even when using transfer learning, i.e., repurposing a

pre-trained model to speed up training and increase accuracy, the creation of a small custom dataset can take allot of time. To reduce or eliminate the need for manually creating datasets it is possible to automatically generated synthetic data. Realistic simulations can be created for this purpose, although creating a high-fidelity simulation may also be time-consuming and expensive. Instead of relying on realism, the simulation can utilize domain randomization (DR) [6-8] to increase the variability of the data, thereby allowing the object detector to learn the important (reoccurring) features that are likely to exist in the real world. It has also been shown that convolutional neural networks (which is an essential part of object detection) are often biased towards texture, and increasing the shape bias can improve their robustness [9]. It is therefore possible that using DR, such as simulating random lighting, camera angles and object positions, can help capture the possible variability in the real world, while the use of random textures and colors on the objects of interest may force the object detector to be more shape biased, thus improving the detection further. The result is a simple simulation that can be used to train an object detection model which is generalized to work on real data. It also simplifies the use of CAD data as realistic renderings can be neglected, especially when the physical appearance of the object can change or deviates from the CAD model.

## 3. Method

### *3.1. Case study: cyber-physical collaboration tool*

The system by [1], introduced in section 2.1 and shown in Fig. 1, was used as the basis for this study, with the environment and objects representing an industrial plant producing offshore products. The physical setup consists of a scaled 2D map of the industrial plant, fitted to a conference table, including movable 3D printed objects, such as buildings, in the same scale. A webcam fixed above the table is used to identify and track the various objects. The tracking system is implemented using the opensource computer vision library (OpenCV) [10] and the object detection (deep learning) framework Darknet [11]. The detected objects are mapped to the four corners of the scaled geographic area using perspective transformation. The new coordinates with object IDs are then sent to the VE developed in Unity through a local network, where their respective 3D models are rendered at the corresponding location in real time and real scale. The VE can be accessed wirelessly through an Oculus Quest VR-headset, enabling a user to explore a believable, real-scale environment, which is considered one of the most important affordances offered by VR [12].

To make the system scalable and easy to implement, our aim is to automatically train the object detection model to identify and locate objects in the real world, using the CAD models as input.

### *3.2. Data acquisition and preparation*

Four different objects in FBX format were included based on their existing CAD models: 3 production buildings and a deepwater jacket construction. The physical scene in Fig. 1. (c) was simulated in Unity, where each object was moved and rotated randomly on the plane through the scripting API. The training datasets were created by capturing images of the scene while running the simulation. The test datasets were created by capturing a video of the physical scene while moving the 3D-printed objects by hand, then extracting some of the frames and labeling them manually. Test samples were taken from various conditions, including different image resolutions, object scales, and room lighting, as shown in the first column in Fig. 2.

Two different detection approaches were tested: one for detecting each object in the scene (4 classes), and the other for additionally detecting the orientation of each object in 36 degrees increments (40 classes, 10 per object). The test set for object detection includes 415 images, and 140 for orientation. Perspective transformation was applied to the train and test datasets for detecting orientation, since the camera position may influence how the orientation of objects are perceived.

### *3.3. Experiment datasets and model architecture*

The 13 different training datasets illustrated in Fig. 2 were created to compare the effect of different data representations and to form an ablation study. Dataset 1 contains a subset of the test data (72 images) to compare the performance between using real and synthetic data. Datasets 2-4 represents a realistic scene, where the background and object textures are known beforehand, including random camera positions (dataset 3) and random lighting angle, intensity, and color (dataset 4). Additional DR were applied (datasets 5-13) by rendering the plane and objects with different shaders in three different conditions: on the plane, on the objects, and on both. Datasets 5-7 were rendered using random images from the Flickr8k dataset [13], to test if textures based on real images influences accuracy. Datasets 8-10 used random textures and colors, and the final datasets 11-13 only used random colors.

Additional data augmentation methods provided by Darknet were used during the training of each model, including random image saturation, exposure, hue, image resizing and network resolutions, in addition to mosaic augmentation and image flipping (not used when detecting rotation angle).

Each synthetic dataset contains 1000 images and was trained for 128 epochs, or 4000 iterations with batch size 32. Transfer learning was used by initiating each training with the

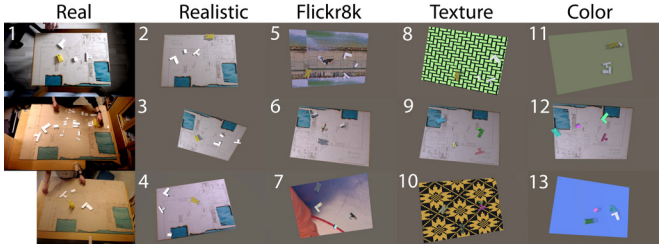


Fig. 2. Overview of the different datasets used in the ablation study, with the numbers referring to specific training datasets and their respective trained models. The left column represents samples from the test dataset.

YOLOv4 weights [11], which is pre-trained on the COCO (common objects in context) dataset. The learning rate was set to 0.001 for the first 80% training iterations and multiplied by 0.1 for each 10% remaining iterations.

Based on the ablation study, a new dataset was created combining the DR techniques that generalizes most to the real data, which is the most relevant approach when assuming the background and object textures are not known beforehand in the real world. Additional model parameters were adjusted when training this dataset in an attempt improve the accuracy, including different learning rates, batch sizes, network resolutions, and adjusting the network for detecting smaller objects and using re-calculated anchors. Additionally, a new dataset was created including distractor objects (random objects without label) to test if it improved the model.

Finally, using the results of the ablation study and model optimization, a model was trained for detecting the orientation of each object in addition to location and object ID.

### *3.4. Evaluation*

The trained models were evaluated using the test datasets to calculate mean average precision (mAP) with intersection over union (IoU) thresholds of 0.5 and 0.75, denoted as mAP@0.50 and mAP@0.75. mAP is a common metric used to evaluate object detection models, which considers both localization and identification of each object for every detection confidence. A higher IoU threshold considers more accurate object localization at the cost of lower detection rates.

The influence of the number of training samples were analyzed using learning curves, showing the change in test accuracy (mAP) based on the amount of generated training images used. This is helpful for determining the tradeoff between training duration (both in terms of generating data and training the model) and model accuracy. Training loss and test accuracy plots are provided to discuss model selection and training duration, and to determine if overfitting occurs.

## 4. Results

### *4.1. Object detection*

The results after training each dataset in the ablation study is presented in Fig. 3. Training on the realistic scene with DR (dataset 4) achieves the best performance with 93.46% mAP@0.50 and 85.75% mAP@0.75, on average 10.4% higher than training with real data (dataset 1). Dataset 8 (random textures on the plane with realistic objects) and 12 (realistic plane with random object colors) scores the highest among the datasets containing generalized DR methods, with mAP@0.50 values of 86.51% and 84.09%, respectively. By combining these results, a dataset was created (8_12) to remove any resemblance to the real scene, where the plane contained random textures and the objects were rendered with random colors. This dataset achieved 86.86% mAP@0.50 and 79.16% mAP@0.75. Adjusting model parameters did not improve the accuracy, except for including distractor objects, which increased mAP@0.75 by 1%.

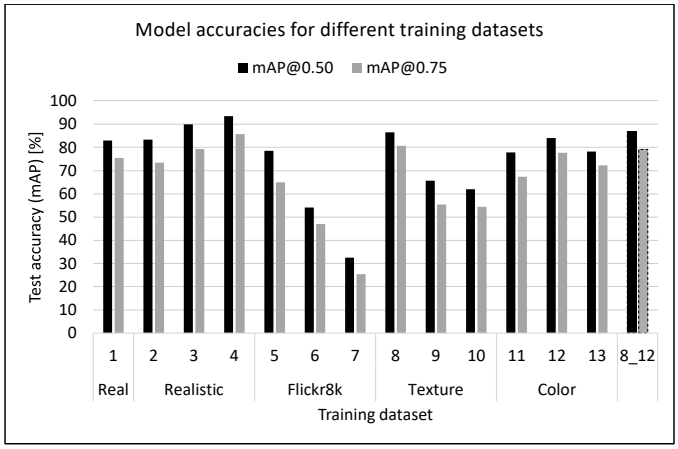


Fig. 3. Results of the ablation study, with test accuracies for each model trained on different dataset. The labels on the horizontal axis refers to Fig.2.

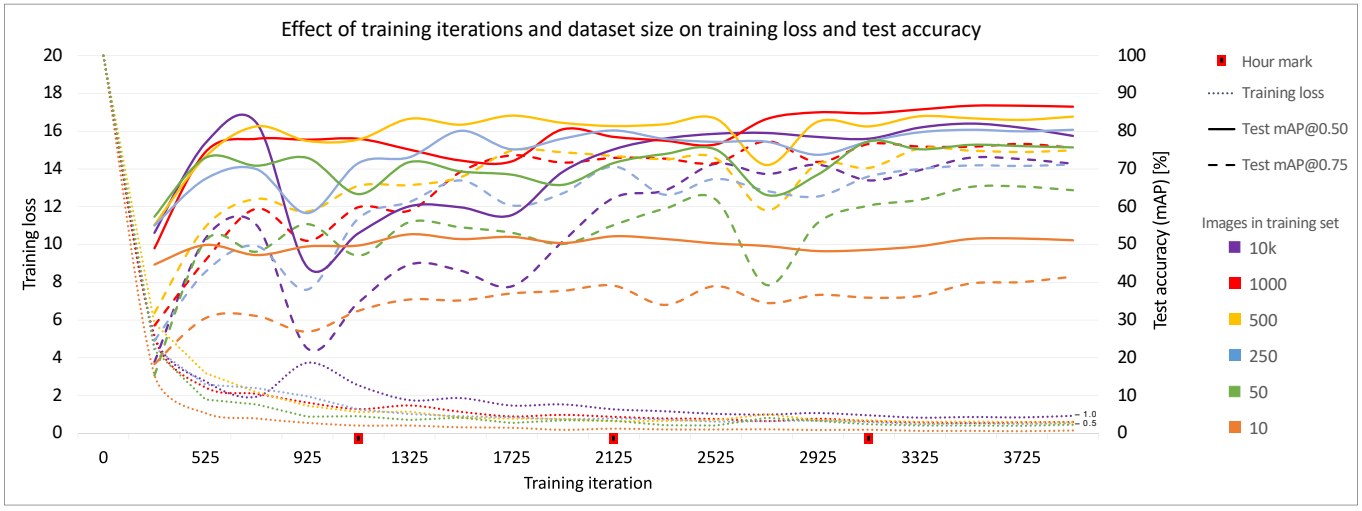


Fig. 4. Loss and mAP values for models trained with different amounts of training data, indicated by color. The calculations are done after several training iterations, with red markers indicating every hour of training.

Fig. 4. shows training loss and test mAP for dataset 8_12, with varying amounts of training images used. Using 1000 training images provided the best model after 4k iterations. Even though the training loss converges early the test accuracy is not stable until the later iterations. mAP values are not decreasing for the test set after several training iterations, thus showing no tendency for overfitting on the training data.

### *4.2. Object detection with rotation*

Based on the ablation study (dataset 8_12), a dataset was generated containing 10k images for detecting the orientation of each object, with 929 images per class on average. The dataset was split into groups of different amounts of images, then trained for 5k iterations each. Fig. 5. shows the resulting mAP for each group, including the results from Fig. 4 after 4k iterations for detecting objects without considering rotation. The highest accuracy for detecting rotation angle was obtained using 2500 images (230 per class), with 76% mAP@0.50 and 72.8% mAP@0.75.

A few example detections from both models are shown in Fig. 6. The middle image on the top row contains multiple incorrect detections due to a low detection threshold (a set confidence value for considering a detection valid) being used. The other images used a detection threshold of 0.95, with only a few miss-classifications such as detecting the hand as one of the objects on the bottom left image.

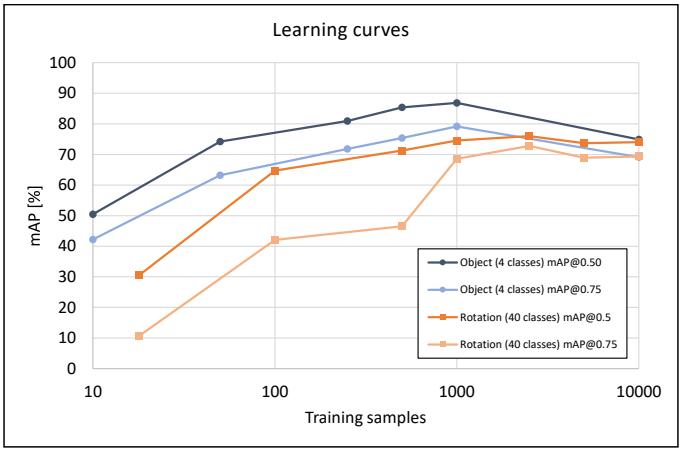


Fig. 5. Learning curves for models trained to detect objects, with and without considering the rotation angle, showing the effect of the number of training samples (log scale) on accuracy (mAP).

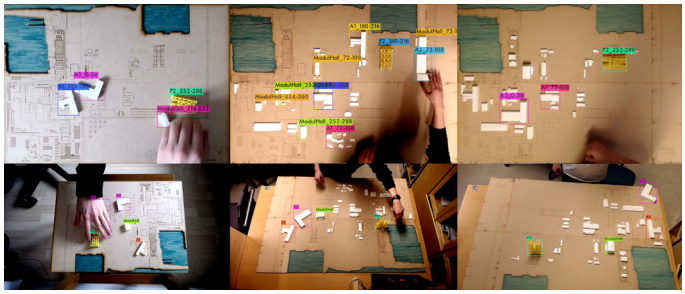

Fig. 6. A few detection samples using real data, with rotation on the top row and simple object detection on the bottom row.

## 5. Discussion

### *5.1. Synthetically trained object detection models*

Using only synthetically generated data from CAD models to enable object detection in the real world has shown promising results. Using the appearance of the real environment as basis for the simulation to generate training data provides the highest accuracy, even better that using real data alone, although the model trained with real data only contained 72 images. However, the effort of creating and preparing a few real training images is higher than generating thousands of synthetic training images automatically. The use of synthetic data is increasingly favorable when including many objects or object states (such as orientation), as the information from the simulation can be used to label many different classes automatically.

Object detection models should also be reliable in various environments, i.e., when the appearance of CAD models and their surroundings are not known prior to production or use-cases. Based on this assumption, we used DR methods to generate training data with random textures and colors, achieving high accuracies while generalizing to real data. Results from Fig. 4. also demonstrates that the model does not overfit on the training data, as there is no noticeable drop in test accuracy with decreasing training loss. It also shows that the training loss (~0.6 in our case) can be used to indicate when a model is properly trained, since creating (real) test data can be a tedious task. However, using test data during training can help pick a model at an earlier stage instead of training for a set number of iterations, as shown in Fig. 4. where most of the models did not improve much after 1725 iterations (roughly 1.6 hours of training). Considering the amount of generated synthetic training data, more is not necessarily better, especially when given a set training duration. Fig. 5. shows that the model does not improve when including more than around 1000 samples for detecting the objects, or 2500 when including rotation, when training each model for roughly 4 hours. However, training with too few samples causes overfitting, with low training loss and poor test accuracy.

### *5.2. Applications*

The system for tracking physical objects and visualizing their configuration in a real-scaled VE can be realized using synthetic data, thus making it scalable and easy to implement. By simply registering a new CAD model, the system can autonomously train the object detection model to track the object in the real world, allowing (non-expert) users to seamlessly add content to the application. The same principle can be used in different contexts where the tracking and identification of physical objects is relevant. Connecting a physical and virtual production environment [4] is an example where the method may be useful, by integrating multiple objects and tracking their movements and states within the real environment. It is also highly relevant for robotic manipulation, to for example locate and handle multiple

objects in an assembly line, without having to manually create training data whenever a new component is introduced.

For applications that require increasingly complex data, where synthetic data alone is insufficient, the synthetically trained model may still be used as a supportive tool for labeling real data through pseudo-labeling. Here, inference on real data is used to detect and label objects, with human input to correct and verify the labels before using the data for training and improving the model.

### *5.3. Limitations*

The presented case only detects objects on a pre-defined plane, which significantly simplifies the problem of real-world localization. The simulation also assumes that the approximate location of the camera relative to the objects is known. For a truly generic approach, the distance between the camera and objects should also be randomized further, allowing objects to be detected at any scale for different applications.

Object detection on images from a single camera does not on its own consider local or global coordinate systems, although promising progress in real-world pose estimation has been made [14-16]. Object detection may in some cases be more useful for giving machines an awareness of their surroundings by being able to identify multiple objects simultaneously with coarse location estimates, while more sophisticated algorithms can be employed for accurate pose estimation for single objects.

Given the few objects included and the specific context of the case study, other applications and domains using different datasets may not produce the same results. The method should therefore be applied and tested in different contexts to further validate the approach.

## 6. Conclusion

Using synthetic data for training object detection models for tracking and identifying objects in the real world has been demonstrated through a case study. The training data was generated by simulating CAD models in a synthetic environment using domain randomization, and the trained models were able to detect the real objects with high accuracies, thus supporting scalable, deep learning-based object detection.

For the cyber-physical VR application, used in the case study, the method supports seamless integration of CAD models to be used in the physical and virtual environments. Exploring and testing the method in different use-cases is needed to further optimize and validate the approach in different contexts.

## Acknowledgements

We thank Peter Bakkeid for providing the concept and digital models for the case study.

## References

[1] Kohtala, S., Kaland, T., Jacobsen, L., Aalto, P. and Steinert, M. Bringing Reality Back to Virtual Reality-A Collaborative Tool for Multidisciplinary Teams. *DS 101: Proceedings of NordDesign 2020, Lyngby, Denmark, 12th-14th August 2020* (2020), 1-12.

[2] Min, X., Zhang, W., Sun, S., Zhao, N., Tang, S. and Zhuang, Y. VPModel: High-fidelity product simulation in a virtual-physical environment. *IEEE transactions on visualization and computer graphics*, 25, 11 (2019), 3083-3093.

[3] Arrighi, P.-A. and Mougenot, C. Towards user empowerment in product design: a mixed reality tool for interactive virtual prototyping. *Journal of Intelligent Manufacturing*, 30, 2 (2019), 743-754.

[4] Damgrave, R. and Lutters, E. Synthetic prototype environment for industry 4.0 testbeds. *Procedia CIRP*, 91 (2020), 516-521.

[5] Mahmoodpour, M., Lobov, A., Hayati, S. and Pastukhov, A. *An Affordable Deep Learning Based Solution to Support Pick and Place Robotic Tasks*. Kalashnikov Izhevsk State Technical University, City, 2019.

[6] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W. and Abbeel, P. *Domain randomization for transferring deep neural networks from simulation to the real world*. IEEE, City, 2017.

[7] Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S. and Birchfield, S. *Training deep networks with synthetic data: Bridging the reality gap by domain randomization*. City, 2018.

[8] Hinterstoisser, S., Pauly, O., Heibel, H., Martina, M. and Bokeloh, M. *An Annotation Saved is an Annotation Earned: Using Fully Synthetic Training for Object Detection*. City, 2019.

[9] Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A. and Brendel, W. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231* (2018).

[10] Bradski, G. The opencv library. *Dr Dobb's J. Software Tools*, 25 (2000), 120-125.

[11] Bochkovskiy, A., Wang, C.-Y. and Liao, H.-Y. M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv:2004.10934* (2020).

[12] Berg, L. P. and Vance, J. M. Industry use of virtual reality in product design and manufacturing: a survey. *Virtual reality*, 21, 1 (2017), 1-17.

[13] Hodosh, M., Young, P. and Hockenmaier, J. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47 (2013), 853-899.

[14] Peng, S., Liu, Y., Huang, Q., Zhou, X. and Bao, H. *PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation*. City, 2019.

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

[16] Tjaden, H., Schwanecke, U., Schömer, E. and Cremers, D. A region-based gauss-newton approach to real-time monocular multiple object tracking. *IEEE transactions on pattern analysis and machine intelligence*, 41, 8 (2018), 1797-1812.