24th Computer Vision Winter Workshop
Friedrich Fraundorfer, Peter M. Roth, Fabian Schenk (eds.)
Stift Vorau, Austria, February 6–8, 2019
DOI: 10.3217/978-3-85125-652-9-02

SyDD: Synthetic Depth Data Randomization for Object Detection using Domain-Relevant Background

Stefan Thalhammer, Kiru Park, Timothy Patten, Markus Vincze
Automation and Control Institute, TU Wien
Gußhausstraße 27-29, 1040 Vienna, Austria
{thalhammer, park, patten, vincze}@acin.tuwien.ac.at

Walter Kropatsch
Institute of Visual Computing and Human-Centered Technology, TU Wien
Favoritenstraße 9, 1040 Vienna, Austria
krw@prip.tuwien.ac.at

Abstract. *In industry CAD-models are readily available while it is expensive to obtain 3D scans of actual objects. Consequently, training object detectors exclusively from CAD-models leads to a considerable decrease of the data creation effort. While this works well for recognition, detection requires better models to distinguish the object of interest from the background and to take the expected sensor properties into account. To tackle this problem we synthetically create depth data with domain-relevant background and apply randomized augmentation to create a superset of the variations of real-world depth images. Results with a state-of-the-art object detector, trained using our synthetic data, show that our approach yields better results than learning from real-world, hand-annotated data with the LineMOD dataset.*

1. Introduction

Assembly systems in manufacturing are subject to increasing number of variants, smaller lot sizes and shorter life cycles. The application of assistance systems will lead to a reduced error rate and increased capacity [4]. The task of visual assistance systems is accurate and robust object detection.

Recently deep learning advanced the state of the art for computer vision tasks such as object detection. While deep networks achieve superior performance, they require a huge amount of training data [8]. Capturing and annotating these data is time and labour consuming and often requires physical instances, which is problematic in fast paced manufacturing environments. Industrial applications, however, typically have CAD-data readily available. We propose to take advantage of this by creating synthetic training data directly from CAD-models by rendering depth images from a virtual scene representing the domain of deployment.

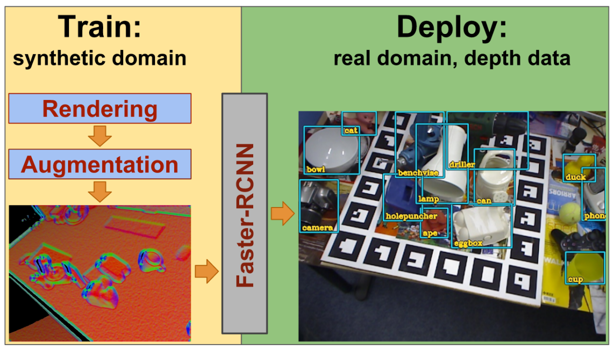

Figure 1. By rendering scenes with domain relevant objects and augmenting the noise model, we create better synthetic training data for object detection.

Synthetic depth images are rendered using CAD models to create a scene then we apply a randomized noise model. A standard tool to create synthetic data is the freely available, open-source software Blender[1] [1, 2, 11]. When training an object detector it is important to create data of sufficient variability to discriminate the objects of interest from the background. For object recognition, where one object is identified in a cropped image, it is known that a randomized background is sufficient to improve re-

[1]www.blender.org

sults [16, 18]. For object detection (i.e. classification and bounding box regression), where multiple objects are identified in a scene, randomized backgrounds are still insufficient to overcome all the ambiguities. Inspired by Handa *et al.* [2], who create full synthetic scenes for a semantic segmentation task, we propose to create scenes that include the expected object placements for better training with realistic depth images.

Another issue to consider is that training deep networks using synthetic data and deploying these on real-world data leads to reduced performance due to the different domains, the so called reality gap. A common method to close the reality gap is to create data of sufficient variability using domain randomization [17] or using the Perlin noise technique [18]. A major challenge is to capture the expected variations in the actual test images. Hence, we propose to combine Perlin noise [10] with a randomized sensor model in order to improve object detection in real-world depth images.

In summary, we propose a domain-related rendering step with an improved noise modelling step referred to as augmentation. Figure 1 outlines the approach. The contributions are the following:

- Rendering synthetic scenes with domain relevant objects to create a realistic background for object detection in depth images.
- Introducing randomized augmentation of synthetic depth images to better capture the expected variations in real-world data.
- Showing advance by evaluating object detection on a standard dataset, the LineMOD dataset [5], since bounding-box targets and class labels are available.

The remaining paper is structured as follows. Section 2 summarizes related work. The rendering and augmentation method is described in Section 3. Section 4 presents the results and evaluation. Section 5 concludes with a short discussion.

2. Related Work

This section discusses synthetic data creation and domain randomization for object detection.

2.1. Synthetic Data Creation

Carlucci *et al.* [1] use *Blender* to create a synthetic depth image dataset for object recognition. They use 3D CAD models downloaded from different databases to create object categories. Views are rendered from a configuration space consisting of object distance, camera position, focal length and random object warping minimizing the amount of identical rendered images. Planche *et al.* [11] present a pipeline to render realistic depth images for object recognition. They simulate the image appearance for a wide range of sensors. Their pipeline consists of a pattern projection mechanism, an intermediate step impinging sensor noise followed by stereo matching and post-processing to reproduce the spatial sensitivity of the sensors and to simulate the impact of surface materials. Backgrounds such as primitive shapes and captured real-world scans can be added. Rozantsev *et al.* [13] project the object geometry, taken from CAD models, into RGB images. A texture filling algorithm varies the object appearance with respect to blur, noise and material properties. Su *et al.* [16] render multiple views of 3D objects to generate a single compact descriptor of that object using a CNN. Handa *et al.* [2] create annotated synthetic indoor scenes using an automatic furniture arrangement mechanism. They use a simulated Kinect noise model to include noise in the synthetic depth scans.

In order to enable object detection in synthetic images it is important to create background information with sufficient variability to separate the objects of interest from the insignificant scene parts. Previous work has only addressed the randomization and augmentation of the generated data for the object of interest, which is mainly due to the focus on the task of object recognition. We instead consider object detection, and thus augment full scenes, including the background, to generate high quality training data.

2.2. Domain Randomization

Since we use an off-the-shelf architecture as detector, trained on synthetic data, it is necessary to transfer the domain to match the real-world image statistics. Domain randomization is a common strategy to create data of sufficient variability to include the variations of a desired domain [15, 17, 18].

Sadeghi *et al.* [15] learn collision avoidance for autonomous flight from simulation. They render RGB-images from synthetic 3D hallways. Parameters such as wall textures, furniture position, illumination and camera pose are randomized. Tobin *et al.* [17] use domain randomization to produce sufficient variability at training time to enable robot

Table 1. Background objects in the virtual scenes.

simple	no additional background information
limited	Apple IMac, bin, keyboard, lamp, laptop, two types of screens, mouse, pot plant, speakers
realistic	All from the *limited* objects, Apple Iphone, ball, BeatsAudio, two types of cans, bottle, Buick model, bulb, DualShock 4 controller, pc fan, knife, Nintendo Gameboy, Nvidia GeForce GTX 1080, plier, spacer, stapler, tablet

grasping. Their approach randomizes shape, position, orientation and texture of the objects involved. The characteristics of lights and the camera extrinsics are also randomized. Zakharov *et al.* [18] use domain randomization to augment depth images. Fractal Perlin noise, Voronoi texturing and white noise is used as background for rendered 3D object models. Perlin noise is an inexpensive way to simulate sensor noise. Randomized patterns are used to simulate occlusion.

A remaining challenge for domain randomization is to randomize the data in the source domain in such a way that the variations of the target domain are captured. We address this by augmenting synthetic depth scans using a combination of Perlin noise and a randomized sensor noise model. Variations of the background information and the occlusion patterns are achieved by randomizing the placement of domain-relevant objects.

3. SyDD: Closing the Reality Gap

We present a method to create and subsequently augment synthetic depth images. The pipeline, named *SyDD*, is presented in Figure 2.

The first step is the creation of synthetic depth images from a virtual scene. The second step is augmenting the synthetic depth images by adding randomly sampled variation to the pixel's depth values. This variation of the augmented domain X^{a} (depth noise, lateral noise, occluded image regions, errors due to the limited depth resolution) creates a superset $X^{\mathrm{a}} \supseteq X^{\mathrm{r}}$ of the variations in real-world scans, i.e., the target domain X^{r}. However, we take care that X^{a} does not diverge too much from X^{r} by choosing variations in a way not to violate the sampling theory.

3.1. Rendering: Synthetic Data Creation

We create synthetic data with diverse scene setups and background information in order to produce data with high variation to train object detectors. Three different approaches to create synthetic scenes are chosen in order to evaluate the importance of the background information:

- *simple*: Objects are arranged on a table, without further background information.
- *limited*: Objects are arranged on a table with static domain-relevant background objects.
- *realistic*: Objects are arranged on a table with static domain-relevant background objects and randomly placed domain-relevant objects.

Table 1 presents a list of background objects used for rendering. The additional objects are downloaded from GrabCAD[2].

For every scene five to eight objects of interest are randomly placed with repetition. These objects are annotated with a bounding box and with pixel class correspondences if fully visible. If not fully visible the bounding box is reduced accordingly and occluded pixels are not annotated. The camera pose is randomly chosen from valid views described in the dataset used for validation. The output of the synthetic data creation step is a depth image, a binary mask indicating visible image regions and a mask indicating pixel level class correspondences. The binary mask provides information about image regions with valid depth values due to the baseline distance of the infrared projector and the sensor. Figure 3 shows an example of the synthetically rendered depth images and visibility masks.

3.2. Augmentation: Randomized Depth Image Variations

We apply an augmentation loosely based on a sensor model and Perlin noise-based pixel warping to the rendered depth images. In order to create a superset of the variations of real-world depth images the parameters of our augmentation are randomly chosen for each image.

Various works evaluate and quantify the errors of the depth scans from infra-red based structured light cameras such as the *Microsoft Kinect V1*. The most common sources of error are the depth sensor itself, the measurement setup and properties of the object surface. Missing depth values are typically caused by infrared occlusion, specular surface reflection and

[2]https://grabcad.com

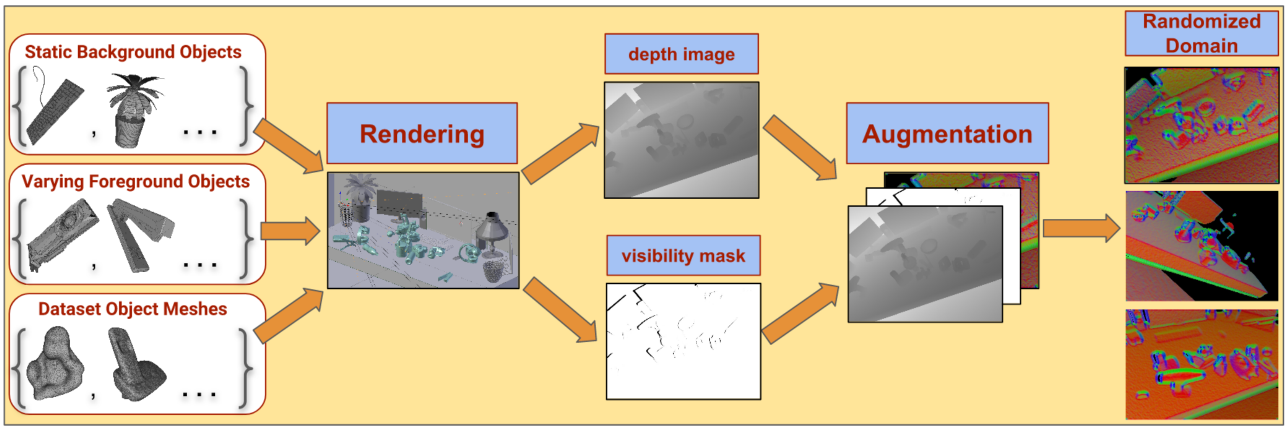

Figure 2. Synthetic depth image creation and augmentation pipeline.

gaps in the depth images due to strong light [7]. Our approach is designed for objects of interest with surface materials that diffusely scatter incoming light, hence omitting the simulation of specular reflections. We propose to randomize the parameters of our augmentation to account for the intractable number of variations and combinations of the influences in the depth image capturing process.

Based on the imaging geometry, parts of the scene are occluded, these occlusions are affected by strong light illuminating the scene. In order to simulate that influence morphological opening and subsequent median filtering is applied to the mask image that is created by the rendering script. The binary mask is applied to the synthetic depth images to remove the occluded image regions. The kernel sizes are sampled from $\{3, 5, 7\}$. These kernel sizes are also used for blurring.

For further augmentation depth images are resized to 320 by 240 pixels, since that is the resolution of the infra-red based structured light camera, the *Microsoft Kinect V1*. The images are down sampled using area interpolation to avoid aliasing. Blur is added to minimize the discrepancy between depth gradients in the real-world and synthetic images. The standard deviation of the blurring operation is chosen uniformly in a range from 0.25 to 3.5. The synthetic depth values are rounded to the nearest quantization value, based on the hypothesized sensor's depth resolution [7] to obtain synthetic depth values in an eleven bit range.

Additional noise is added to the quantized depth values using an offset chosen randomly from a Gaussian distribution. The depth noise of the sensor increases non-linearly with depth, though since the expected object placement is in a range of 65 centimeters to 115 centimeters we approximate it linearly, similar to [6]. The offset is calculated per pixel using its nearest quantized value, scaled by the parameter n_{sd}. The randomized parameter n_{sd} is drawn uniformly between 0.002 and 0.004. This range is based on the actual depth noise of the *Microsoft Kinect V1*.

Further randomness of the appearance of occluded scene parts, depth and lateral noise is added by warping the depth images through the application of pixel offsets, using the Perlin noise technique. This approach is similar to Zakharov *et al.* [18]. The basic concept is that a 3D vector field is generated to randomly distort synthetic depth images. Pixel locations are warped by applying the sampled vector field to the already augmented depth images. We use their proposed parameter ranges.

An example of the synthetic depth scans is presented in Figure 4.

4. Experiments

Three experiments are conducted to evaluate our approach. First, we compare object detection trained on the same number of real-world and on synthetic depth images. Second, results are presented providing quantitative information about the influence of the background in the synthetic scene. Third, the influence of the different steps of the augmentation method is shown. Finally, we discuss open problems.

All the experiments are conducted on the LineMOD dataset [5], which is taken from the *SIXD Challenge 2017*[3]. This is a standard and well-known baseline for object recognition and pose estimation in RGB-D. The test set of the LineMOD dataset consists of 15 test sets, one for each dataset object, with approximately 1200 captured images per scene. Every set has different object instances visible, although

[3]http://cmp.felk.cvut.cz/sixd/challenge_2017

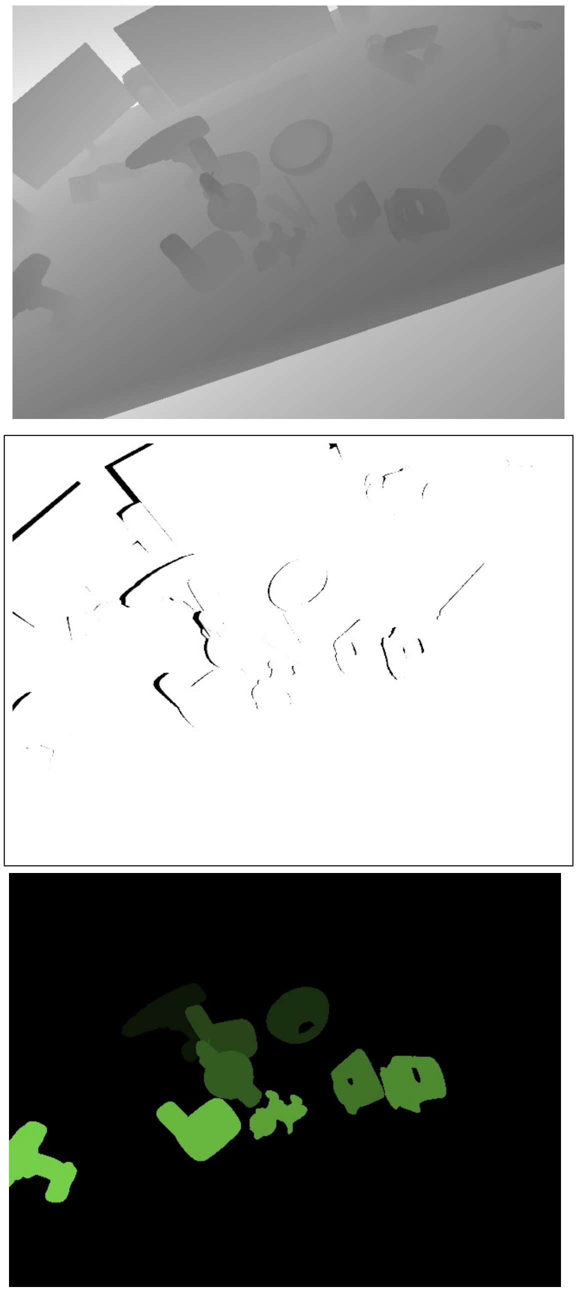
Figure 3. Synthetic depth image (top), visibility mask (middle) and pixel level class correspondence (bottom).

only the object in the center of the image is annotated with a bounding box, class and pose. An exception is the *benchvise* test set that has all dataset objects annotated. Since different object instances without annotation are visible in the test images, only the annotated object is considered for calculating the detection recall. In all experiments we report the percentage of correctly detecting and classifying annotated objects with an Intersection-over-Union (IoU) of 0.5.

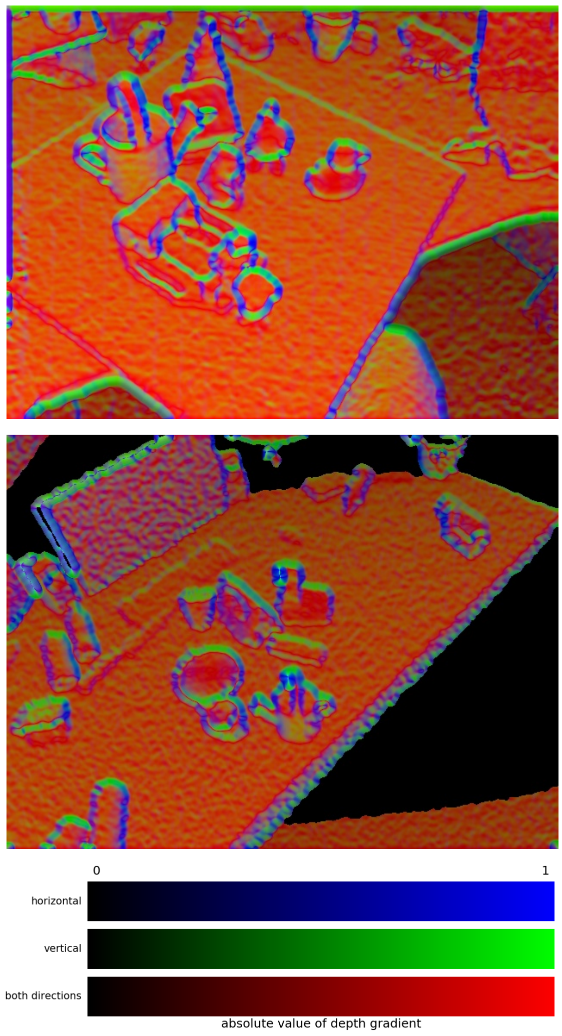

Figure 4. Comparison of a real-world (top) and a synthetic (bottom) depth image, converted to RGB images.

4.1. Experimental Setup

All tests are conducted with the following preprocessing and network configuration. All real-world and synthetic images are converted to three channel RGB images. These are coloured based on the normal direction using the approach of Nakagawa *et al.* [9]. Image regions with missing depth values are inpainted and depth cuts are applied up to 20 centimeters and regions further than 1.8 meters.

We use Faster-RCNN [12] with ResNet-101 [3] backbone, pretrained on ImageNet [14], with the standard optimizer and loss functions. The learning rate starts at 0.01 and decays to 0.0001. We train for 180000 iterations using a batch size of one and a weight decay of 0.0001.

4.2. Performance on real-world data

We compare an object detector trained on real-world images that are taken from the *benchvise* test

Table 2. Detection recall of Faster-RCNN trained on real-world and on synthetic data. Numbers in percent.

Classes	real	*SyDD*
ape	53.56	**76.86**
can	**97.24**	94.15
cat	41.31	**82.70**
driller	**96.21**	92.76
duck	89.39	**93.70**
eggbox	64.8	**81.01**
glue	**81.72**	70.08
holepuncher	**89.89**	77.69
overall	76.77	**83.62**

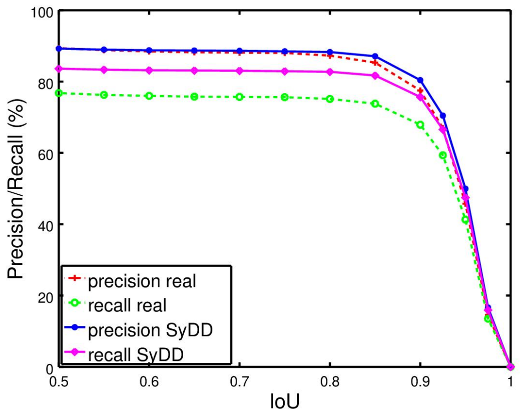

Figure 5. Recall and precision curve comparison of real-world and synthetic data using different IoU scores.

set of the LineMOD dataset against an object detector trained on images created by *SyDD*. Table 2 compares per category detection recall.

The average recall of the detector trained on our synthetic dataset outperforms the recognizer trained on real-world data. The performance margin results from the higher variability in the synthetic dataset. The biggest differences in detection recall are visible for the objects *ape*, *cat* and *eggbox*. This is caused by the scene setup used for capturing the real-world depth scans. The *ape* is placed in different poses in the scene and is either not occluded or completely occluded in most of the images. The *benchvise* test set only includes these extreme cases and does not have many examples for partial occlusion. The *cat* is placed with the same pose in all scans, which again results in very low visibility or full visibility with the addition of low variability of pose in the *benchvise* test images. The *eggbox* is placed in different poses but with a strong similarity of viewpoints. Furthermore, occlusion is caused mostly by the same object. The randomized augmentation covers a wider range of variations influencing the image creation process

Table 3. Detection recall of Faster-RCNN trained on *SyDD*, with different backgrounds in the virtual scenes. Numbers in percent.

Classes	simple	limited	realistic
ape	57.79	59.79	**79.69**
benchvise	65.16	64.09	**96.05**
bowl	91.24	**93.03**	85.81
camera	66.61	74.94	**94.17**
can	57.02	79.10	**91.97**
cat	58.95	80.75	**97.71**
cup	76.13	83.06	**88.06**
driller	65.99	84.76	**96.72**
duck	82.30	81.58	**95.37**
eggbox	83.32	92.42	**93.77**
glue	65.16	79.98	**82.79**
holepuncher	74.54	85.53	**92.97**
iron	42.19	64.58	**89.84**
lamp	50.77	65.69	**96.09**
phone	71.36	68.22	**93.00**
overall	63.03	72.29	**85.88**

as well as the placement of objects in the virtual scene. This increases the variation of occlusions and views in comparison to the real-world images in the *benchvise* test set. Figure 5 indicates that the performance of the Region Proposal Network is not affected by the usage of our synthetic training data.

4.3. Influence of the Background Information

The importance of the background information in the training data is evaluated by comparing three object detectors trained on different background objects, each consisting of 10000 images. Table 3 shows the performance recall.

The results indicate that the usage of additional background information during synthetic data generation improves the detection recall. Results also indicate that is unnecessary to use the same background objects during training as during deployment. Our findings show that domain specific background objects are sufficient for detectors to yield similar performance to detectors trained on real-world, hand-annotated images. The reader is directed to the detection results of the *bowl*. The recall for this object decreases with the usage of more comprehensive background information.

4.4. Evaluation of the Augmentation Method

The influence of the augmentation used for creating training data is evaluated by comparing four object detectors trained on 10000 images. The augmen-

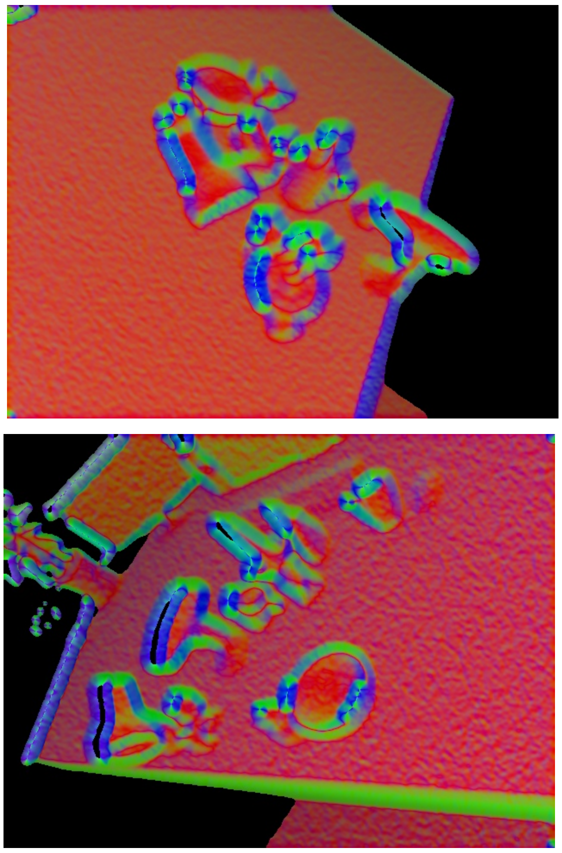

Figure 6. Exemplary images, displaying the synthetic training dataset, with simple background information (top) and limited background information (bottom).

tation methods are:

- *synth*: non augmented synthetic depth data.
- *perlin*: augmenting the synthetic images only using Perlin noise with the parameters from [18], after removing occluded image regions using the randomized visibility mask.
- *auth*: randomized realistic sensor model, where the difference to our proposed method *SyDD* is that the depth noise n_{sd} is added before quantizing these to eleven bit range.
- *SyDD*: our proposed augmentation.

The results presented in Table 4 indicate that strong average detection performance is achieved when adding Perlin noise. However, even better performance is achieved using our augmentation. We conclude that augmenting images with Perlin noise can effectively close the domain gap, but combining with a randomized sensor model leads to even more powerful detectors. Re-sampling the augmented images to the *Kinect's* depth resolution decreases detection and classification results.

Table 4. Detection recall of Faster-RCNN trained using different augmentation methods. Numbers in percent.

Classes	*synth.*	*perlin*	*auth.*	*SyDD*
ape	59.06	71.76	67.96	**79.69**
benchvise	71.99	93.90	91.85	**96.05**
bowl	**91.64**	91.48	91.08	85.81
camera	56.54	84.60	89.84	**94.17**
can	53.51	94.15	**95.32**	91.97
cat	89.91	97.20	91.18	**97.71**
cup	73.23	84.19	81.21	**88.06**
driller	89.31	95.62	95.71	**96.72**
duck	62.04	93.78	89.63	**95.37**
eggbox	45.49	81.80	90.90	**93.77**
glue	44.02	**85.08**	83.44	82.79
holepuncher	59.18	**93.37**	81.33	92.97
iron	39.15	89.41	78.83	**89.84**
lamp	75.31	93.48	**97.96**	96.09
phone	45.78	91.31	90.27	**93.00**
overall	59.76	83.82	82.28	**85.88**

4.5. Open Problems

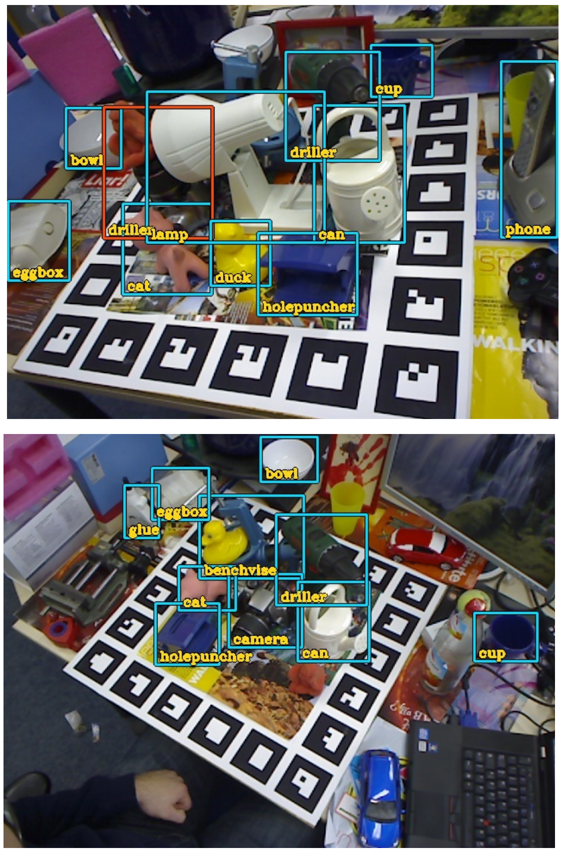

Figure 7. Detection result with incorrect detections of stacked objects.

Qualitative results of object detection using training images from *SyDD* and test images from LineMOD are presented in Figure 7 and Figure 8.

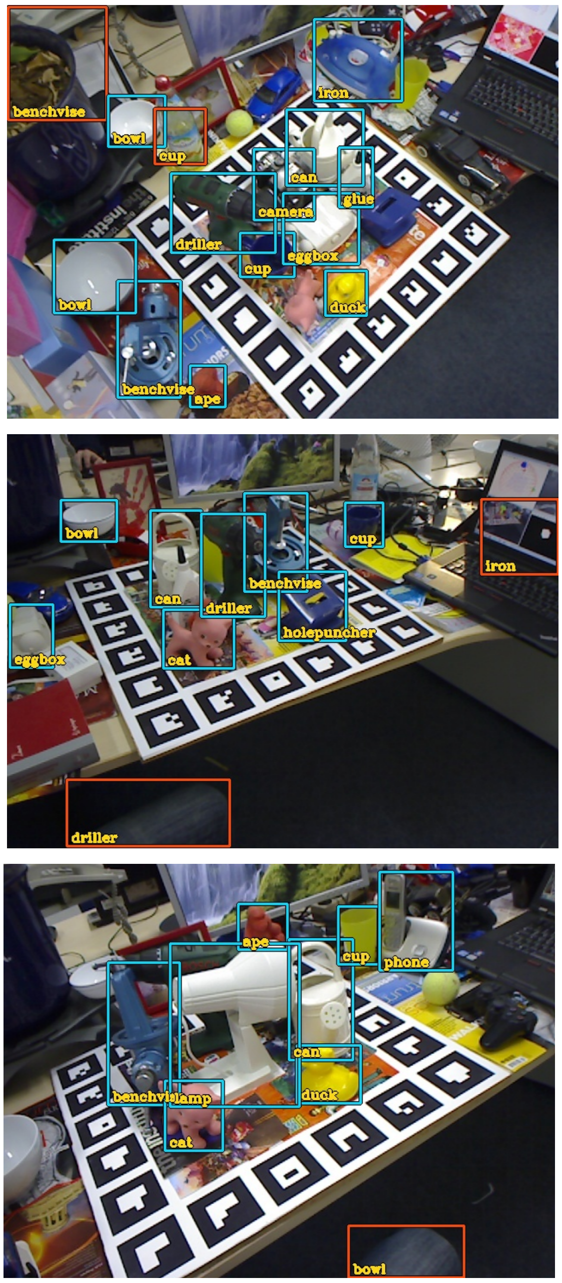

Figure 8. Detection result with incorrect detections on boundary regions of the image or fabric.

The RGB-images are only used for visualization. The top of Figure 7 shows an *ape* placed on top of a *camera*, which is incorrectly classified as *driller*. A similar error is visible in the bottom of Figure 7. The *benchvise* is correctly classified but the *duck* is not detected. This error arises because objects in the virtual scene are enclosed by a convex hull. Consequently, perfectly stacked objects can not be found in the synthetic images. A convex hull is used to represent the collision shape of objects to minimize errors when performing the physics simulation.

Figure 8 shows detection results with objects incorrectly detected on fabric, near the image boundary. The top image shows an incorrect detection of *benchvise* in the upper left corner of the image. Another incorrectly detected instance of *iron* is visible in the middle image on the right edge. These detections result from annotating only partly visible objects that are cropped by the image boundary during training. Another common error is the detection of objects on smoothly curved fabric surfaces as can be seen in the bottom parts of the middle and the bottom image in Figure 8. This error is a combination of annotating boundary regions in the synthetic images and missing background information during the rendering process.

5. Conclusion

We present a pipeline to create and augment synthetic depth data to close the reality gap for object detection. Our experiments demonstrate that deep networks trained using our data outperform detectors trained on available real-world, hand-annotated data. This is promising because we can significantly reduce the time and effort to generate training data for real-world deployment of modern computer vision algorithms.

Our method efficiently closes the domain gap on the LineMOD dataset and hence completely alleviates the need to use real-world training data. The main drawbacks of our method can be easily overcome by fine tuning the rendering script to the desired task, but would compromise the generalization of the approach. We show that the usage of domain-specific objects creates discriminating background information for object detectors trained using synthetic data. Additionally we show that it is preferable to use an augmentation loosely based on a sensor rather than using an authentic sensor model.

Our domain randomization approach omits certain aspects of depth image variations since they are not relevant for the challenges at hand. Future work will address the task of generalizing our domain randomization to other sensors.

Acknowledgements

This work has been supported by the Austrian Research Promotion Agency in the program "Production of the Future" funded project MMAssist_II (FFG No.: 858623) and the Austrian Ministry for Transport, Innovation and Technology (bmvit).

References

[1] F. M. Carlucci, P. Russo, and B. Caputo. A deep representation for depth images from synthetic data. In *Robotics and Automation, IEEE International Conference on*, pages 1362–1369. IEEE, 2017. 1, 2

[2] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla. Understanding real world indoor scenes with synthetic data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4077–4085, 2016. 1, 2

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[4] S. Hinrichsen, D. Riediger, and A. Unrau. Assistance systems in manual assembly. In *Proceedings 6th International Conference on Production Engineering and Management*, pages 3–13, 2016. 1

[5] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer, 2012. 2, 4

[6] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. *IEEE Winter Conference on Applications of Computer Vision*, 2017. 4

[7] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12:1437–1454, 2012. 4

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[9] Y. Nakagawa, H. Uchiyama, H. Nagahara, and R.-I. Taniguchi. Estimating surface normals with depth image gradients for fast and accurate registration. In *3D Vision, International Conference on*, pages 640–647. IEEE, 2015. 5

[10] K. Perlin. Improving noise. In *ACM Transactions on Graphics*, volume 21, pages 681–682. ACM, 2002. 2

[11] 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 *3D Vision, International Conference on*, pages 1–10. IEEE, 2017. 1, 2

[12] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pages 1137–1149, 2017. 5

[13] A. Rozantsev, V. Lepetit, and P. Fua. On rendering synthetic images for training an object detector. *Computer Vision and Image Understanding*, 137:24–37, 2015. 2

[14] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015. 5

[15] F. Sadeghi and S. Levine. CAD2RL: Real single-image flight without a single real image. In *Robotics: Science and Systems*, 2017. 2

[16] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 2

[17] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Intelligent Robots and Systems, IEEE/RSJ International Conference on*, pages 23–30. IEEE, 2017. 2

[18] S. Zakharov, B. Planche, Z. Wu, A. Hutter, H. Kosch, and S. Ilic. Keep it unreal: Bridging the realism gap for 2.5d recognition with geometry priors only. *2018 International Conference on 3D Vision*, pages 1–11, 2018. 2, 3, 4, 7