Synthetic Data of Randomly Piled, Similar Objects for Deep Learning-Based Object Detection

Janis Arents[1(✉)], Bernd Lesser[2], Andis Bizuns[1], Roberts Kadikis[1], Elvijs Buls[1], and Modris Greitans[1]

[1] Institute of Electronics and Computer Science, 14 Dzerbenes St., Riga 1006, Latvia
{janis.arents,andis.bizuns,roberts.kadikis,elvijs.buls,modris_greitans}@edi.lv
[2] Virtual Vehicle Research GmbH, Inffeldgasse 21a, 8010 Graz, Austria
bernd.lesser@v2c2.at
http://www.edi.lv/en, https://www.v2c2.at

Abstract. Currently, the best object detection results are achieved by supervised deep learning methods, however, these methods depend on annotated training data. With the synthetic data generation approach, we intend to mimic the real data characteristics and diversify the dataset by a systematic rendering of highly realistic synthetic pictures. We systematically explore how different combinations and portions of real and synthetic datasets affect object detectors performance. The developed synthetic data generation framework shows promising results in deep learning-based object detection tasks and can supplement real data when the variety of real training data is insufficient. However, when synthetic data ratio increases over real data ratio, a decrease in average precision can be observed, which has the most affect on 0.75-0.95 IoU threshold range.

Keywords: Synthetic data · Deep learning · Object detection

1 Introduction

Even though we have almost spent a decade since the term Industry 4.0 was introduced [16], industrial robots by themselves still mostly have limited intelligence. One of the most popular ways to make robots smarter is by attaching cameras [4] that can acquire information on a specific region of interest or perceive the environment around the robot and adjust their movements respectively [19]. Therefore, by giving the robots an ability to *see*, comprehend and act accordingly, we can automate tasks that traditionally require human intelligence or complex and very spacious machines [5].

Specific computer vision problems - object detection and instance segmentation - are two of the main prerequisites to automate a number of tasks where industrial robots must proceed in an unstructured environment. The detection

S. Sclaroff et al. (Eds.): ICIAP 2022, LNCS 13232, pp. 706–717, 2022.
https://doi.org/10.1007/978-3-031-06430-2_59

indicates where in the camera's frame an object is located, and which class does this object belong to. Whereas segmentation determines which class does every pixel of an image belongs to. Instance segmentation is a type of segmentation that differentiates among pixels belonging to different instances of the same class. With this information, one can acquire a visible shape of a specific object and use it to determine objects pose [12], which in turn is handy for picking up and manipulating the object. However, the segmentation task typically is computationally more expensive than object detection or classification tasks [9], therefore in the case of randomly piled objects it is rather inefficient to segment all of the objects in the region of interest. In this article, we focus on detecting objects that have the highest possibility to result in a successful grasp.

Object detection is hard in the case of randomly piled objects. The objects are often only partially visible, and when the pile consists of similar or even the same objects, the similar features that could be used to detect the unobstructed objects are scattered all over the pile. There are still challenges to retaining low false-positive rates in cluttered environments [14]. In such environments robotic grasping is hardly competing with human performance, therefore many manipulations still require manual work or rather big and expensive machines that are hard to adjust if product assortment changes.

Currently, in case of known and finite amount of instances, the best object detection results are achieved by supervised deep learning methods, for example EfficientDet [25] and HRNet-OCR [29]. However, these methods depend on annotated training data. Each new object requires numerous new training examples of pile images, and labelling of such images is a time consuming manual labour, especially in the case of image segmentation tasks. To alleviate the training data acquisition process and simplify the use of modern computer vision methods in industry, we turn to data synthesis.

Image synthesis or rendering is a process of generating digital images from virtual scenes. The photo-realism of rendered images, videos, and computer games keeps increasing. Also, the tools for creating virtual environments with included physics simulation are becoming more user-friendly and affordable. For example, such tools as Blender, Unity, and Unreal Engine can be used free of charge. Therefore AI and computer vision research community increasingly use such tools to generate data, train systems in virtual environments directly, and to adjust and develop more vision-oriented tools, for example, an open-source plugin UnrealCV [20].

The main goal of this article is to explore and develop a synthetic data generation framework for object detection tasks. We compare manually labelled data with synthetically generated data and analyze how does the real data, synthetic data and different combinations of both affect the precision of object detection models. Taking into account the above mentioned the rest of the paper is structured as follows: Sect. 2 describes state-of-the-art developments in the fields of object detection and data synthesis. Section 3 describes our proposed method. In Sect. 4 several different tests are performed and the obtained results are analyzed. Section 5 concludes this article.

2 Related work

Object detection in images has improved significantly in the last years [31]. Current best detectors are based on deep neural networks and trained in a supervised fashion. Notable detectors include types of Region-Based Convolutional Neural Networks (Faster R-CNN [24], Mask R-CNN [13], Beta R-CNN [28]). Even though the mean average precision (mAP) is an important aspect to compare between object detectors, other aspects such as computational efficiency, memory consumption and inference time also play an important role in selecting an appropriate object detector for the intended use-case. In this sense, the YOLO [22,23] variants provide better leverage between the precision and speed when compared to others [31].

Detection and segmentation of objects that are randomly piled combined with an industrial robot pick and place operation [3] are commonly referred to as bin-picking. Even though this problem has been addressed by the research community for several decades, it remains one of the most challenging tasks in robotics [2,6]. Multiple instances of the same type of objects that are randomly stacked in a pile introduce a variety of challenging conditions, such as the similarity between foreground objects and the background, occlusions, and clutter. That in a combination with sensor noise contribute to the complexity of object detection [4].

Most of the research is focused on applications that have publicly available large data-sets on commonly used objects, whereas in industrial applications object types can be specific to the respective product. Model-free grasping techniques partly addresses data absence [17] and can proceed without having prior knowledge of the objects, but this method complicates the post-gripping [33] process and can introduce additional steps for precise positioning. The latest achievements in the field of computer-generated imagery widen opportunities in synthetic real-life like data generation for object detection tasks in this particular scenario.

Synthetic data for object detection tasks typically has been composed by placing foreground objects on background scenes with different parameters that can be varied. Some approaches proceed with 2D images that are placed on a set of background images [7,11], with set of rules or physical simulation for piling the objects realistically. The level of complexity for such and similar methods is lower, however they are typically restricted to 2D nature that contributes to the lack of realism, which therefore decreases detectors performance. 3D graphical engines have also been utilized in synthetic data generation, however, these are typically dedicated for household situations or usage with every day objects [26,27].

A comprehensive overview of the existing state of the art datasets relevant for object pose estimation for industrial bin-picking is given in [15]. Even though we don't address pose estimation task in this article, this overview gives a valuable insights in the current progress and the applicability of the available data sets. It shows that most previous approaches are not well suited to be directly used

for deep learning methods in bin-picking task due to either a lack in the amount of data, its variety or incomplete ground truth information.

Even though modern object detection methods are effective at detecting of traffic signs [32], pedestrians [10], household objects [8] etc., the methods fall short in industrial settings due to lack of training data. Our proposed method utilizes modern graphics engines to generate realistic data for training neural networks for robotic grasping in industrial specific scenarios, where data can be use-case specific and change over time. We focus on generating versatile data by a vast amount of adjustable parameters. Thus the parameters can be adjusted by specific needs. In this case, we extract the data of unobstructed objects in corresponding orientation to train AI models only on the objects that have the highest probability to result in a successful grasp when the model is deployed. The object detection described in this article serves as the first step in the whole pick and place process, and only information about the most promising objects that could be grasped will be processed further.

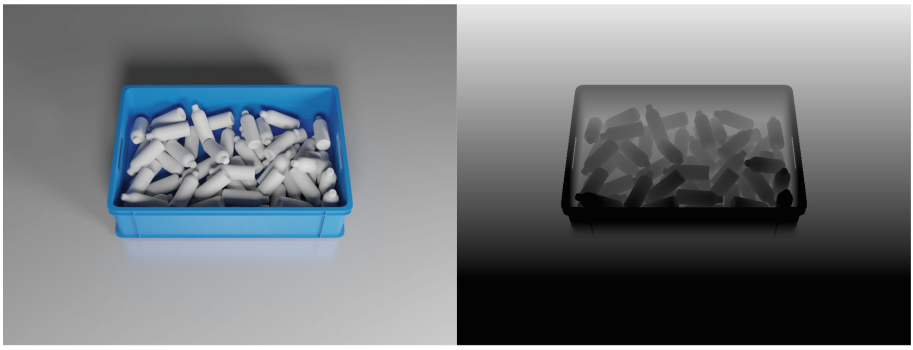
With respect to the synthetically generated data, in this work, we create a new dataset fully suitable for deep learning by the means of the amount of data and sufficient ground truth annotations. Although we currently feature only one 3D model, we aim at synthesizing photo-realistic images by means of higher resolutions, materials, textures, lighting, reflections and indirect light bounces instead to reduce the reality gap.

3 Proposed Method

3.1 Synthetic Generation of Realistic Pile Images and Annotations

We aim at automating the systematic rendering of highly realistic synthetic pictures to generate data sets for training the object detection algorithm. The image generator obtains images by arranging any kind of objects that have 3D models within a virtual 3D scene from which it renders highly realistic images. In this case, it is a box with white bottles. By tuning parameters of the 3D scene such as an object, camera and light positions, object colour or texture and surface properties, brightness, contrast, and saturation, a large image diversity can be generated in resolution and levels of realism depending on the user needs. By further exporting relevant ground truth data from the 3D scene including the location and orientation of objects within a rendered image, the generated data is fully annotated.

By sampling all possible configurations of objects and image parameters within the 3D scene (in arbitrary, user-defined granularity), a modification space is defined allowing for the automated generation of large synthetic data sets, for which the diversity of images is controlled by the user. By systematically defining appropriate scenes and modification spaces, the image generator can be used to generate not only training and validation data sets in sufficient quantity (overcoming a lack of training data, which is often a limiting problem), but in general, also allows for generating data sets specifically designed for specific experiments



(a) Generated image (b) Generated depthmap

(c) Ground truth bounding box (d) Ground truth segmentation image

Fig. 1. Synthetic data rendered with Blender

Fig. 2. Different perspectives and light conditions

later on (e.g. to compare across different classification models or to extract specific insights of the algorithm).

In this work, we focus on automating the generation of realistic data sets for training our object detection algorithm, i.e. highly realistic images of piled objects together with their corresponding depth-map, for which we annotate every single object visible in the scene with location and orientation ground truth. Corresponding instance-segmented images are also generated as depicted in Fig. 1d. For image generation, we use the free open-source tool, for which we fully automated the rendering of parametrizable 3D scenes (as they have been defined within the scope of this work) in *python* by making use of blender's python API.

We implemented a basic set of functions in python for setting up virtual 3D scenes and controlling the image rendering process, where we kept the configuration of objects and image parameters for the defined configuration space as variables. These functions have been implemented in such a way, that the whole process of image generation iterates across all possible configurations of varying image parameters within the given configuration space (i.e. the placement of the 3D objects into different positions of the 3D scene, applying different textures and shaders to the objects, different positioning of light sources and the camera, running the rendering process) is being automated and executed in headless mode and no manual interaction via Blender's GUI is required.

To additionally allow for a more realistic object placement within the considered 3D scenes, especially for piling up any number of objects more realistically and naturally, we further make use of Blender's integrated physics engine, which enables us to optionally apply simple rigid body simulations based on convex hull collision detection. For postprocessing and annotation of images, we further make use of OpenCV, which we make available to blender's internal python interpreter via pip.

3.2 Training Setup

A wide variety of machine learning algorithms are currently available to detect objects in images. At the beginning of the study, YOLOv3 [23] was one of the most popular real-time, single-stage object detection algorithms with the best AP scores and FPS trade-offs [31]. In addition, YOLOv3 can be used with a variety of frameworks [1,30] and has a large community to discuss important issues. We used YOLOv3 for all following experiments

To train and evaluate the YOLOv3 models, we used the Darknet [1] framework and already pre-trained convolutional weights [21] *darknet53.conv.74* on Imagenet. This framework is a branch of the original Darknet open-source framework [21] with various improvements such as anchor calculation or charts. Darknet framework is implemented using the *c++* programming language and is efficient at training other popular neural networks as well. Training and evaluation was performed on two separate *Linux* OS workstations with *Nvidias RTX 3090* and *A100* GPUs.

4 Tests and Results

The performed tests are structured in a way to evaluate how object detector trained on synthetic datasets performs in real-life scenarios compared to object detector trained on real datasets. Even though the main focus is concentrated on synthetic data usage, we also systematically explore how different combinations and portions of real and synthetic datasets affect object detectors performance. In respect to the use-case, the main goal is to detect objects that are most likely to result in a successful grasp, therefore we also analyze in how many scenes at least one object has been found above a certain Intersection over Union (IoU) threshold.

4.1 Datasets

Real Datasets. The real data was gathered by randomly distributing bottles in a box. For each of the acquired images, the positions of the bottles were altered. The intensity of lightning and camera exposure time was systematically modified to acquire a high diversity of different lighting conditions. In total for training purposes, 2200 real images were acquired and manually labelled from which 1760 images for training and 440 images for validation purposes. Furthermore, the training and validation datasets were rotated four times by 90°, in total resulting in 7040 training images and 1760 validation images.

Two test datasets were gathered and manually labelled, first dataset *Test 1* was captured in similar conditions as the real training dataset, however *Test 2* was captured with different camera and in different conditions.

Synthetic Datasets. For the experiments considered in this work, we have generated a synthetic dataset in the same amount as the real dataset, consisting of 8800 photo-realistic high-resolution scenes. For every individual scene we fill an initially empty box with randomly placed bottles by making use of Blender's physics simulation engine to achieve realistic positioning and orientation. We use realistic textures and Blender's principled BSDF shader nodes to achieve realistic renderings of the scenes including lighting, reflections and indirect light bounces.

After filling the box with the bottles, we create a series of renderings for which we vary power levels of 4 different light sources and the orientation of the camera, which orbits around the box and renders the scene from 16 different angles as depicted in Fig. 2. For every camera orientation, we also generate a depth image and the segmentation images of the individual bottles as seen by the camera and labelled by the object ID as illustrated in Fig. 1. We further generate a separate annotation file for every camera perspective which contains the individual object's orientation and rotation in-camera coordinates together with the object's visibility percentage.

4.2 Evaluation Metrics

The most common metric used to measure the accuracy of the object detection in the images is average precision (AP) [18], which is also utilized in this article to evaluate the performance of object detection in our experiments. In our case the Darknet framework with variously set (0.5–0.95) IoU thresholds is used to perform AP measurements. With the IoU we measure the overlapping area between the ground truth and the predicted bounding box and evaluate the precision over different thresholds. True positive (TP), false positive (FP), and false negative (FN) estimates for each detected object are used to calculate precision and recall to perform AP measurements.

4.3 Comparisons

The performance and precision of the proposed synthetic data generation approach were evaluated by various aspects. First, the object detection performance was investigated with deep learning models trained using different ratios of synthetic and real data combinations and by utilizing the maximum amounts of the acquired data. Starting with 100% of real data, the real data ratio was incrementally decreased by 10% at the time by substituting it with the synthetic data as depicted in Table 1.

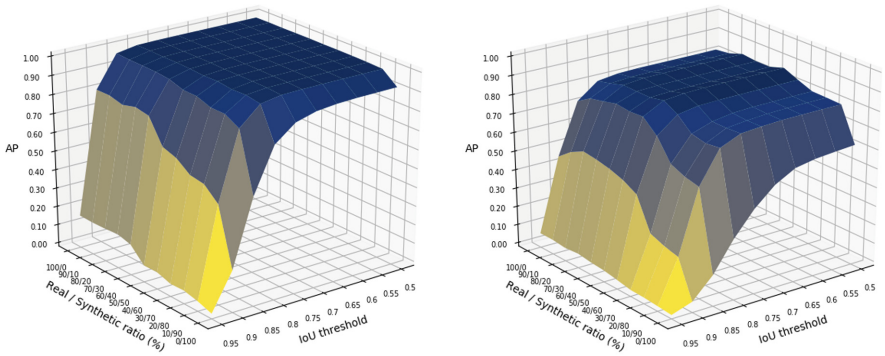
Table 1. Evaluation of object detectors precision

Data distribution			Test 1			Test 2		
Real	Synthetic	Real/Synthetic Ratio %	**Step**	AP @0.5:0.95	OD %	Step	AP @0.5:0.95	OD %
8800	0	100/0	9100	88.61	100.00	9100	69.22	96.95
7920	880	90/10	7900	88.61	100.00	9200	71.04	98.47
7040	1760	80/20	6000	88.36	100.00	8000	73.23	100.00
6160	2640	70/30	6300	**88.65**	100.00	7600	72.83	100.00
5280	3520	40/60	6900	88.22	100.00	7400	72.50	100.00
4400	4400	50/50	4400	85.82	100.00	5000	**73.84**	100.00
3520	5280	40/60	8000	85.59	100.00	8700	70.27	100.00
2640	6160	30/70	7200	84.39	100.00	5900	64.57	100.00
1760	7040	20/80	7200	84.23	100.00	4500	63.62	100.00
880	7920	10/90	8200	82.59	100.00	7000	63.25	100.00
0	8800	0/100	7700	70.01	100.00	5000	38.66	100.00

The evaluation was performed on the described datasets - *Test 1* and *Test 2*. Object detector evaluated on data close to real training data *(Test 1)* scored similar average precision results when real data amount was higher than synthetic data as depicted in Fig. 3a. This also holds true to higher IoU threshold values from 0.85 to 0.95. All of the trained models showed similar average precision results in the IoU threshold region from 0.5–0.8. The main difference can

be seen in the case when the model is trained on purely synthetic data, as the precision remarkably decreases.

A different situation can be seen when the object detector is evaluated on a test dataset that contains different environmental parameters - *Test 2*. In this case the synthetic data supplements real data and increases average precision, whilst achieving the highest precision on a 50/50 ratio. Similarly as with the evaluation results on *Test 1* dataset, also in this case object detector trained on purely synthetic datasets showed the least precision.



(a) Test 1 (b) Test 2

Fig. 3. Average precision of object detection models on real images over different IoU thresholds, viewed by the ratio of real to synthetic data in the training datesets

Even though the object detector trained on real data or different combinations outperforms the detector trained on purely synthetic data, the main precision aspects in this article are connected to the specific use-case. Respectively, the goal is to detect at least one object in the scene with an IoU threshold above 0.95. Obtained results on this aspect are depicted in Table 1 under columns Object Detected (OD). On both test datasets, the trained models could meet this requirement, except in Test 2 case, when the model was trained on purely real data and in following 90/10 ratio.

5 Conclusion

In dynamic environments, especially in the case of randomly piled objects, a lot of uncertainties and different environmental conditions can be present. Ideally, these different conditions should be covered by the training data set in a deep learning-based object detection task to satisfy the precision requirements. However, gathering and labelling the real data is a tedious task and requires a certain amount of human resources and in some cases, it is complicated to recreate all the possible configurations. With the synthetic data generation approach, we intend

to mimic the real data characteristics and diversify the dataset by a systematic rendering of highly realistic synthetic pictures. By tuning image parameters such as an object, camera and light positions, object colour or texture and surface properties, brightness, contrast, saturation, a large image diversity can be generated in resolution and level of realism depending on the requirements. In this article we explore the usage of synthetic data in this particular scenario with one object type, however, the generator can arrange any kind of objects that have 3D models.

The generated dataset, real dataset and different combinations of both were used to train an object detector. The trained models were evaluated on two different test datasets. In most of the cases when the real data ratio was higher than synthetic data, the model achieved higher precision ratings. Even though the models trained on purely synthetic images has lower average precision on real images, the achieved precision is sufficient in our use-case. Thus, by diversifying the training dataset with synthetic images a precision increase can be observed, however when the synthetic ratio is over 50%, the precision decreases.

The developed synthetic data generation framework shows promising results in deep learning-based object detection tasks and can supplement real data when the variety of real training data is insufficient. However, adding the synthetic data to real data requires testing to find the peak of precision, as adding more synthetic images results in lower precision. In respect to future work, the synthetic data generator will be further improved and utilized in object segmentation tasks.

Acknowledgements. The work has been performed in the project AI4DI: Artificial Intelligence for Digitizing Industry, under grant agreement No 826060. The project is co-funded by grants from Germany, Austria, Finland, France, Norway, Latvia, Belgium, Italy, Switzerland, and Czech Republic and - Electronic Component Systems for European Leadership Joint Undertaking (ECSEL JU). In Austria the project was also funded by the program "IKT der Zukunft" of the Austrian Federal Ministry for Climate Action (BMK). Parts of the publication was written at Virtual Vehicle Research GmbH in Graz and partially funded within the COMET K2 Competence Centers for Excellent Technologies from the Austrian Federal Ministry for Climate Action (BMK), the Austrian Federal Ministry for Digital and Economic Affairs (BMDW), the Province of Styria (Dept. 12) and the Styrian Business Promotion Agency (SFG). The Austrian Research Promotion Agency (FFG) has been authorised for the programme management.

References

1. AlexeyAB: darknet. https://github.com/AlexeyAB/darknet. Accessed 20 Dec 2021
2. Alonso, M., Izaguirre, A., Graña, M.: Current research trends in robot grasping and bin picking. In: Graña, M., López-Guede, J.M., Etxaniz, O., Herrero, Á., Sáez, J.A., Quintián, H., Corchado, E. (eds.) SOCO'18-CISIS'18-ICEUTE'18 2018. AISC, vol. 771, pp. 367–376. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-94120-2_35

3. Arents, J., Cacurs, R., Greitans, M.: Integration of computervision and artificial intelligence subsystems with robot operating system based motion planning for industrial robots. Autom. Control Comput. Sci. **52**(5), 392–401 (2018)
4. Arents, J., Greitans, M.: Smart industrial robot control trends, challenges and opportunities within manufacturing. Appl. Sci. **12**(2), 937 (2022). https://doi.org/10.3390/app12020937
5. Arents, J., Greitans, M., Lesser, B.: Artificial Intelligence for Digitising Industry, Applications, chap. Construction of a Smart Vision-Guided Robot System for Manipulation in a Dynamic Environment, pp. 205–220. https://www.riverpublishers.com/book_details.php?book_id=967, https://doi.org/10.13052/rp-9788770226639 (2021)
6. Buchholz, D.: Bin-Picking - New Approaches for a Classical Problem. Ph.D. thesis, ̃(Jul 2015), https://publikationsserver.tu-braunschweig.de/receive/dbbs_mods_00060699
7. Buls, E., Kadikis, R., Cacurs, R., Ārents, J.: Generation of synthetic training data for object detection in piles. In: Eleventh International Conference on Machine Vision (ICMV 2018). vol. 11041, pp. 533–540. International Society for Optics and Photonics, SPIE (2019). https://doi.org/10.1117/12.2523203, https://doi.org/10.1117/12.2523203
8. Chu, F.J., Xu, R., Vela, P.A.: Real-world multiobject, multigrasp detection. IEEE Robot. Autom. Lett. **3**(4), 3355–3362 (2018)
9. Das, A., Kandan, S., Yogamani, S., Krizek, P.: Design of real-time semantic segmentation decoder for automated driving (2019)
10. Dollar, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. IEEE Trans. Pattern Anal. Mach. Intell. **34**(4), 743–761 (2011)
11. Dwibedi, D., Misra, I., Hebert, M.: Cut, paste and learn: Surprisingly easy synthesis for instance detection. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 1310–1319 (2017). https://doi.org/10.1109/ICCV.2017.146
12. Gupta, S., Girshick, R., Arbeláez, P., Malik, J.: Learning rich features from RGB-D images for object detection and segmentation. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8695, pp. 345–360. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10584-0_23
13. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of the IEEE international conference on computer vision, pp. 2961–2969 (2017)
14. He, R., Rojas, J., Guan, Y.: A 3D object detection and pose estimation pipeline using RGB-D images. In: 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1527–1532. IEEE (2017)
15. Kleeberger, K., Landgraf, C., Huber, M.F.: Large-scale 6D object pose estimation dataset for industrial bin-picking (2019)
16. Luenendonk, M.: Industry 4.0: definition, design principles, challenges, and the future of employment (2019). Accessed 24 2020
17. Mousavian, A., Eppner, C., Fox, D.: 6-dof graspnet: variational grasp generation for object manipulation (2019)
18. Padilla, R., Netto, S.L., da Silva, E.A.B.: A survey on performance metrics for object-detection algorithms. In: 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 237–242 (2020). https://doi.org/10.1109/IWSSIP48289.2020.9145130
19. Pérez, L., Rodríguez, Í., Rodríguez, N., Usamentiaga, R., García, D.F.: Robot guidance using machine vision techniques in industrial environments: a comparative review. Sensors **16**(3), 335 (2016)

20. Qiu, W., Yuille, A.: UnrealCV: connecting computer vision to unreal engine. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 909–916. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_75
21. Redmon, J.: Darknet: open source neural networks in c. http://pjreddie.com/darknet/ (2013–2016)
22. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271 (2017)
23. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
24. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (06 2015). https://doi.org/10.1109/TPAMI.2016.2577031
25. Tan, M., Pang, R., Le, Q.V.: Efficientdet: scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10781–10790 (2020)
26. Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., Birchfield, S.: Deep object pose estimation for semantic robotic grasping of household objects. In: CoRL (2018)
27. Wang, K., Shi, F., Wang, W., Nan, Y., Lian, S.: Synthetic data generation and adaption for object detection in smart vending machines. CoRR abs/1904.12294 http://arxiv.org/abs/1904.12294 (2019)
28. Xu, Z., Li, B., Yuan, Y., Dang, A.: Beta R-CNN: Looking into pedestrian detection from another perspective. In: NeurIPS (2020)
29. Yuan, Y., Chen, X., Wang, J.: Object-contextual representations for semantic segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12351, pp. 173–190. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58539-6_11
30. YunYang1994: tensorflow-yolov3. https://github.com/YunYang1994/tensorflow-yolov3 Accessed Dec 20 2021
31. Zaidi, S.S.A., Ansari, M.S., Aslam, A., Kanwal, N., Asghar, M., Lee, B.: A survey of modern deep learning based object detection models. In: Digital Signal Processing, p. 103514 (2022)
32. Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., Hu, S.: Traffic-sign detection and classification in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2110–2118 (2016)
33. Zoghlami, F., Kurrek, P., Jocas, M., Masala, G., Salehi, V.: Design of a deep post gripping perception framework for industrial robots. J. Comput. Inf. Sci. Eng. **21**, 1–14 (2020). https://doi.org/10.1115/1.4048204