Visual Data Simulation for Deep Learning in Robot Manipulation Tasks

Miroslav Surák[1], Karel Košnar[2](✉), Miroslav Kulich[2], Viktor Kozák[2], and Libor Přeučil[2]

[1] Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic
surakmir@fel.cvut.cz

[2] Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague, Prague, Czech Republic
{karel.kosnar,miroslav.kulich,viktor.kozak,libor.preucil}@cvut.cz
http://imr.ciirc.cvut.cz

Abstract. This paper introduces the usage of simulated images for training convolutional neural networks for object recognition and localization in the task of random bin picking. For machine learning applications, a limited amount of real world image data that can be captured and labeled for training and testing purposes is a big issue. In this paper, we focus on the use of realistic simulation of image data for training convolutional neural networks to be able to estimate the pose of an object. We can systematically generate varying camera viewpoint datasets with a various pose of an object and lighting conditions. After successful training and testing the neural network, we compare the performance of network trained on simulated images and images from a real camera capturing the physical object. The usage of the simulated data can speed up the complex and time-consuming task of gathering training data as well as increase robustness of object recognition by generating a bigger amount of data.

Keywords: Deep learning · CNN · Simulation · Ray-tracing · Robot manipulation · Random bin picking

1 Introduction

Robotic manipulators are used in an industrial automation for decades. Typical use-cases vary from welding to pick-and-place tasks. Nowadays, the cooperative robots share the workspace with humans and therefore traditional approaches, relying on precise predefined positions of items in robots workspace, are not working anymore. The robot needs to sense its working space with different sensors and adapt its actions according to the actual situation. With recent progress in deep learning, there start attempts to solve situations, where the robot needs to grasp an object in a random position with end-to-end neural networks trained from large training datasets.

https://doi.org/10.1007/978-3-030-14984-0_29

The deep convolutional neural networks (CNNs), especially when working with images, need a huge amount of labeled data to train. Getting data with proper labels from the real world is usually time-consuming, and often a manual task. For example, this end-to-end network approach [10] makes use of RGB-D sensor and more than 50 thousands of grasping trials and needs 700 h of robot labor. Therefore, there is a need to speed up and automation of data collecting and labeling.

One possible way is to use simulated images for the training of the CNN. However, training from synthetic images can lead to overfitting of the network to 'unrealistic' details only present in synthetic images and failing to generalize well on the real images. The use of a simulator as realistic as possible is a way presented in this paper.

1.1 Related Works

Grasping movement is typically planned directly from RGB or RGB-D image of target objects. Analytic approaches register actual data to the database of 3D models of known objects with precomputed grasping points [1,6,14]. A registration often involves many intermittent steps like image segmentation, classification and pose estimation, where each step typically depends on multiple parameters, that are difficult to tune.

Very good results with utilizing simulated data of 3D point clouds achieves the approach described in [7] for defining grasping points. It achieves better results than analytic approaches. [8] introduces the extension of previous for suction cups grasping.

Alternative approaches are making use of deep learning to estimate the 3D pose of the object directly from intensity image and/or 3D point cloud [5,15]. As there is a need for a large number of training data, a new approach is to train the network on simulated images [13] and to adapt the representation to real data [12]. The work [3] improves the precision of recognition by adding synthetic noise to synthetic training images. Recent research suggests that in some cases it may be sufficient to train on datasets generated using perturbations to the parameters of the simulator [4].

2 Problem Definition

The problem solved in this paper is motivated by the real-world problem of picking of specific metallic parts of a single type from a transportation package and feed these to an automated industrial assembly line. As this task is highly repetitive and the motion performed by the human worker is tedious and one-sided, there is a request for automation.

Parts are not fully randomly distributed in the package, as they are originally organized in columns, but get scattered during the transport. It is expected by the end-user, that manipulator can pick more than 80% of the object from the package. The assembly line needs one part every 60 s. Another request is

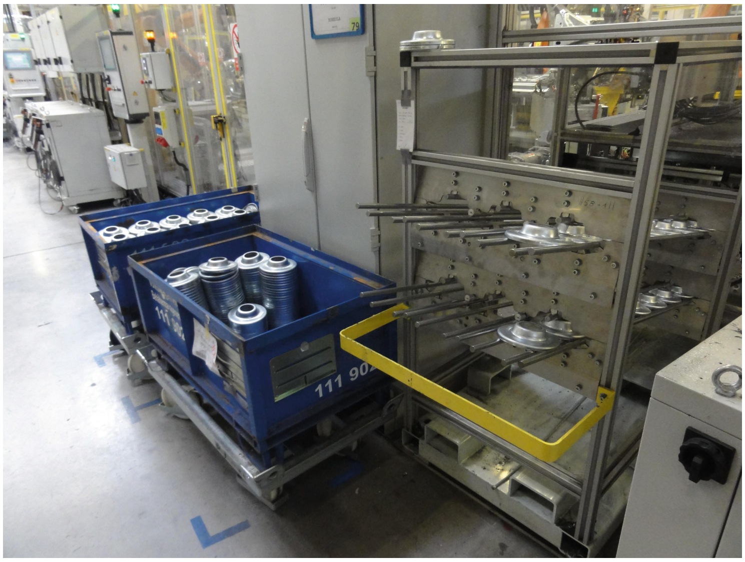

Fig. 1. Image of the transportation package with objects and feeder of the automatic assembly line

flexibility of the solution, as there are many different types of parts manually feed to automated assembly lines. Therefore, modification of the solution for the new part should be as easy as possible (Fig. 1).

As the existing solution described in the following section is based on convolutional neural network, it needs a huge amount of training data. Therefore, this paper focuses on the generation of simulated training data and evaluation of the usage of this data in the described solution.

3 Solution

Pose estimation of the objects for picking is described in details in diploma thesis [11]. Pose estimation of the object position is divided into three steps: segmentation of the image and detection of regions that contain a single object, raw estimation of the position and accuracy improvement.

3.1 Segmentation

The segmentation of the object is base on the Histogram of Oriented Gradients (HOG) approach [2] with the sliding window. This segmentation method was used because is easy to train and performs well under different conditions, e.g. change of light. The parameters of the HOG detector are: block size 16×16, cell size 8×8, image patch size 64×64. A simple SVM classifier is used for classification if the window contains an object or not. Image patches detected as containing object by HOG are used in later steps of the algorithm.

3.2 Raw Estimation of Pose

The size and the position of the center of the patch segmented in the previous step by HOG are used as a first estimation of the distance and position of the object respectively. This first estimation is not accurate enough for reliable picking the object from the box. Therefore next step is necessary to improve the estimation of the position to the level, where the gripper can reliably pick the object. Moreover, the orientation (normal vector) of the object is necessary to estimate to allow successful picking of the object.

3.3 Accuracy Improvement Using CNN

For the further improvement of the object position accuracy, the deep convolution network (CNN) is used. For the needs of the CNN, the previously detected patches are resized to the unified size of 64×64 pixels. As the image patches are resized to unified size, it is not possible for the CNN to directly estimate the position and distance of the object and only multiplicative coefficient of the position in x-y plane and distance from the previous step are trained.

The input of the network is an image patch of size 64×64 pixels. It is followed by four convolutional layers with ReLU activation function followed by max pool layers. Usage of max pool layers effectively decreases the number of parameters of the model, because of the sparsity of data. The last layer of the network is a fully-connected layer with 3 neurons, whose output are predicted position coefficients. The network is learned by the back-propagation approach.

4 Gathering of Training Dataset

The gathering of training data is a semi-autonomous process. At first, the precise position of the learned part is determined by manually placing the gripper on the part. Then, the gripper with the camera is automatically placed into predefined positions in different distances and angles. As the position of the part in the transportation package, e.g. at the bottom, on the top or near the package wall, influence the appearance of the part, this procedure is repeated with part placed in the different configuration in the transportation package. For each configuration are gathered hundreds of images.

These images were then processed with HOG detector and only image patches that contains the part are used in later steps. Also, the relative position between the camera and the part was calculated in this step from the original position of gripper placed on the part and actual position of the gripper with the camera.

The training data consist from: (1) truth relative position between the camera and the object and (2) gray-scale image patch.

4.1 Synthetic Training Dataset

To be able to train CNN from synthetic training data, we need to obtain the same data in the same format. The most crucial part is the gray-scale image.

As the technical drawing of the part is available, it was easy to get the 3D model of the part in question. Now, the realistic gray-scale image of the model with proper lighting, shading, and reflection is necessary to simulate. As the most promising approach seems to use ray-tracing software. This software can realistically simulate all the complicated reflections and lighting of 3D models with different materials and textures. Our choice is to use the Persistence of Vision Raytracer (PoV-Ray) [9] (see Fig. 2 for example of the result) as it is open-source and authors are familiar with the usage of this software.

Fig. 2. Example of the result of PoV-Ray.

The real placement of the camera is in the center of the gripper head with circular light around the camera. See Fig. 3 depicting gripper head with the camera, sucking cups and circular light. Therefore, it was necessary to simulate the camera with the same field of view and the same light source around the camera, to get the same reflections on the surface of the parts. The object and the camera was placed in the same position as gathered by real manipulator with real part. So the synthetic dataset is as near to real one as possible.

The next task was to find the correct material and texture of the model, that is as near to appearance of the real part as possible. The similarity was evaluated by human eyes and improved in an iterative way to achieve the results depicted in Fig. 4.

5 Experiments Description and Evaluation

In the experiments, we compare the errors of the estimated position of the parts. We create two training datasets of the same size of 1000 images. The first dataset

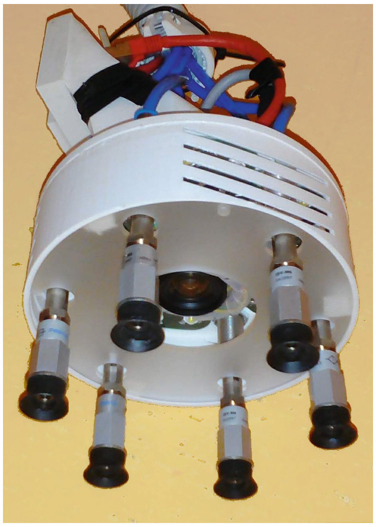

Fig. 3. Configuration of the gripper head with centered camera, suction cups and circular light.

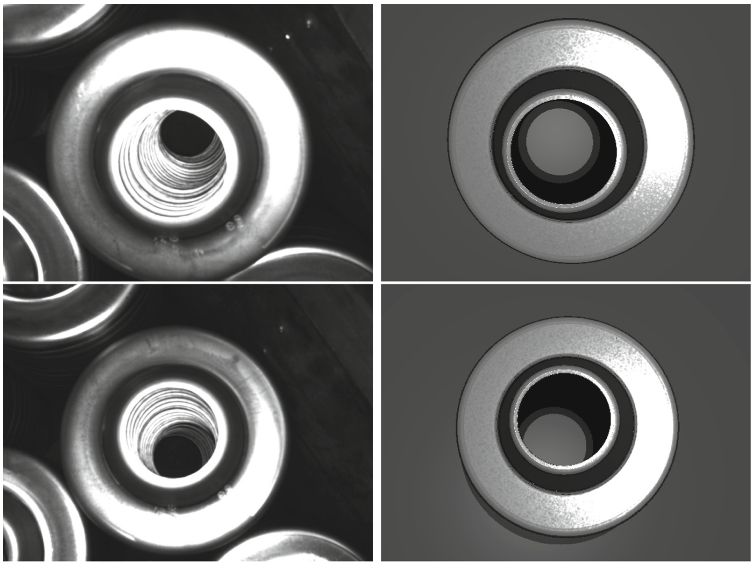

Fig. 4. Example of the real and corresponding simulated images.

was collected with the real camera placed on the real manipulator. The second dataset was generated in the PoV-Ray software. Both datasets contain the same items, the images taken from same positions with the same lighting.

Also, we create a testing dataset with 200 images. The testing dataset was collected with the real camera on the real manipulator.

Two networks were trained in the supervised-learning fashion using the Mean Squared Error. Adam optimizer with learning rate 0.001 was used to find the optimal weights. The training required 5000000 iterations, the dropout rate of 0.5 was used. The first network was trained on the real dataset and the second network was trained on the synthetic dataset.

To get the reference performance, the first network trained on real training dataset was run on the real testing dataset (see Fig. 5). The achieved errors where used as a reference point for the comparison. The performance of the second network ran on the real testing dataset (see Fig. 6) is compared with the first one.

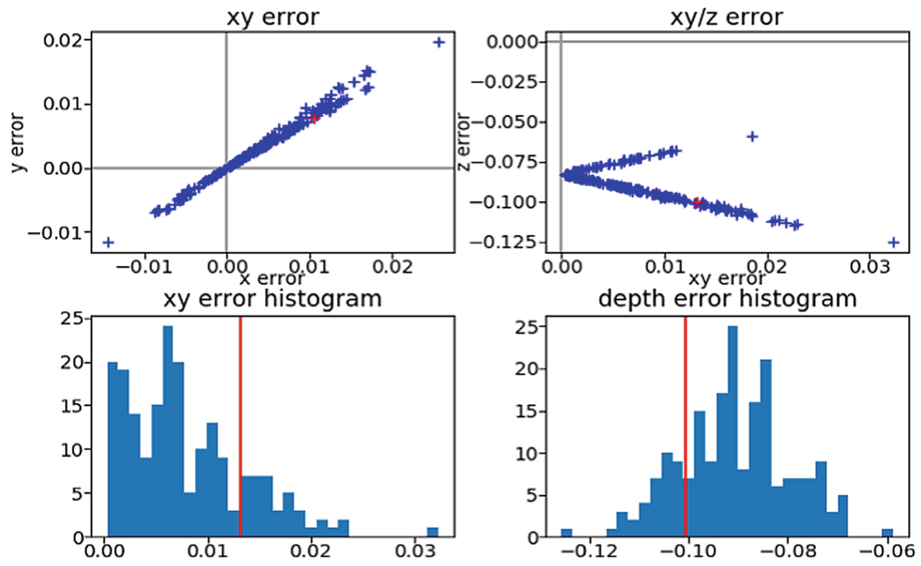

Fig. 5. Errors in position for network trained and tested on real dataset.

The results of the network trained on the synthetic dataset are slightly worse than the original network trained on the real images. The difference between the two networks are less than 10% and that is in the tolerance for the deployment into the real process. The precision of the position determination is in average 7% worse. The precision of the depth determination is in average 3.5% worse. The variance of the position and depth errors are not significantly worse.

The time needed for the collection of the real dataset with 1000 images is around 1.5 h. The synthetic dataset of the same size can be generated on the MetaCentrum Grid Infrastructure in order of minutes.

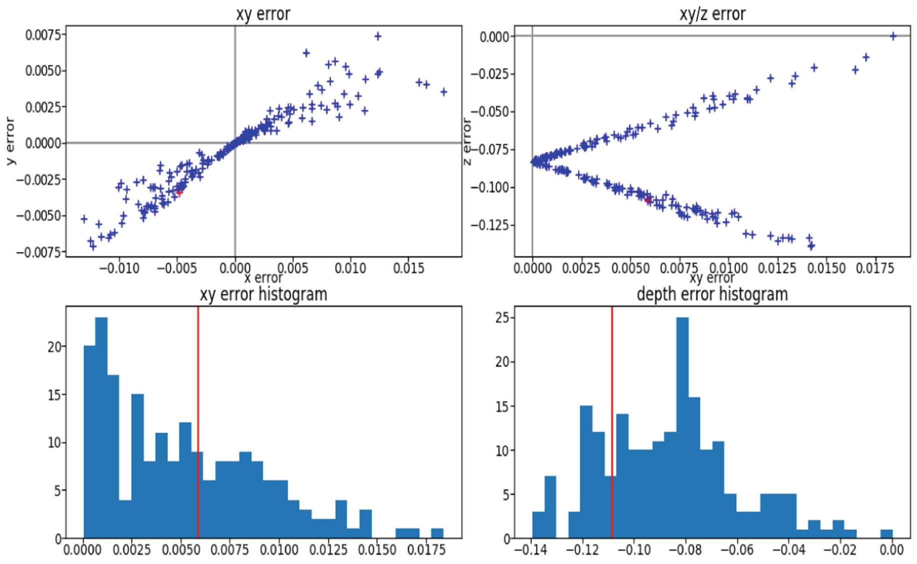

Fig. 6. Errors in position for network trained on synthetic dataset and tested on real dataset.

6 Conclusion and Further Work

The performance of the network trained on the synthetic dataset is slightly worse than the network trained on the real dataset, but the difference is in the tolerance, so the network trained on the synthetic dataset can be deployed with the real manipulator.

Now used part is quite simple and rotational symmetrical, so we can use a quite a small dataset for training. As we plan to use this system for more complex parts, there will be a need for the much bigger dataset and then the time savings will be more significant.

For further improvement, we plan to combine the real and synthetic data together to improve the performance of the network. Also, we plan to replace the manual tuning of the material parameters in the ray-tracing software with automated process of learning the parameters from the performance of the network. As the material parameters significantly influence the light reflections, it is expected, that with the better estimation of material parameters, the simulated images will be more realistic.

Acknowledgement. Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated. This work has been supported from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688117 (SafeLog). The work was also supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS18/206/OHK3/3T/37.

References

1. Ciocarlie, M., Hsiao, K., Jones, E.G., Chitta, S., Rusu, R.B., Şucan, I.A.: Towards reliable grasping and manipulation in household environments. In: Khatib, O., Kumar, V., Sukhatme, G. (eds.) Experimental Robotics. Springer Tracts in Advanced Robotics, vol. 79. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-28572-1_17
2. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proceedings of 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005 (2005). https://doi.org/10.1109/CVPR.2005.177
3. Eitel, A., Springenberg, J.T., Spinello, L., Riedmiller, M., Burgard, W.: Multimodal deep learning for robust RGB-D object recognition. In: IEEE International Conference on Intelligent Robots and Systems (2015). https://doi.org/10.1109/IROS.2015.7353446
4. Gualtieri, M., Pas, A.T., Saenko, K., Platt, R.: High precision grasp pose detection in dense clutter. In: IEEE International Conference on Intelligent Robots and Systems (2016). https://doi.org/10.1109/IROS.2016.7759114
5. Gupta, S., Arbeláez, P., Girshick, R., Malik, J.: Aligning 3D models to RGB-D images of cluttered scenes. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2015). https://doi.org/10.1109/CVPR.2015.7299105
6. Hinterstoisser, S., et al.: Multimodal templates for real-time detection of textureless objects in heavily cluttered scenes. In: Proceedings of the IEEE International Conference on Computer Vision (2011). https://doi.org/10.1109/ICCV.2011.6126326
7. Mahler, J., et al.: Dex-Net 2.0: deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In: Robotics: Science and Systems (2017). https://doi.org/10.15607/RSS.2017.XIII.058, http://arxiv.org/abs/1703.09312
8. Mahler, J., Matl, M., Liu, X., Li, A., Gealy, D., Goldberg, K.: Dex-Net 3.0: computing robust robot vacuum suction grasp targets in point clouds using a new analytic model and deep learning. In: 2018 IEEE International Conference on Robotics and Automation (IRCA) (2018). https://doi.org/10.1109/ICRA.2018.8460887
9. Persistence of Vision Pty. Ltd.: POV-Ray - the persistence of vision raytracer (2004). http://www.povray.org/
10. Pinto, L., Gupta, A.: Supersizing self-supervision: learning to grasp from 50K tries and 700 robot hours. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 3406–3413, June 2016. https://doi.org/10.1109/ICRA.2016.7487517
11. Sushkov, R.: Detection and pose determination of a part for bin picking. Master thesis, Czech Technical University in Prague, June 2017. https://dspace.cvut.cz/handle/10467/68468?show=full
12. Tzeng, E., et al.: Adapting deep visuomotor representations with weak pairwise constraints. In: The 12th International Workshop on the Algorithmic Foundations of Robotics (2016)
13. Varley, J., Weisz, J., Weiss, J., Allen, P.: Generating multi-fingered robotic grasps via deep learning. In: IEEE International Conference on Intelligent Robots and Systems (2015). https://doi.org/10.1109/IROS.2015.7354004

14. Xie, Z., Singh, A., Uang, J., Narayan, K.S., Abbeel, P.: Multimodal blending for high-accuracy instance recognition. In: IEEE International Conference on Intelligent Robots and Systems, pp. 2214–2221 (2013). https://doi.org/10.1109/IROS.2013.6696666
15. Zeng, A., et al.: Multi-view self-supervised deep learning for 6D pose estimation in the Amazon Picking Challenge. In: Proceedings of IEEE International Conference on Robotics and Automation (2017). https://doi.org/10.1109/ICRA.2017.7989165