Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 56-2 (2023) 5233–5239

Sim2Real Grasp Pose Estimation for Adaptive Robotic Applications

Dániel Horváth [*,**] Kristóf Bocsi [*] Gábor Erdős [*,***]
Zoltán Istenes [**]

[*] Centre of Excellence in Production Informatics and Control, Institute for Computer Science and Control, Eötvös Loránd Research Network, Budapest, Hungary (e-mail: daniel.horvath@sztaki.hu)
[**] CoLocation Center for Academic and Industrial Cooperation, Eötvös Loránd University, Budapest, Hungary
[***] Department of Manufacturing Science and Engineering, Budapest University of Technology and Economics, Budapest, Hungary

Abstract: Adaptive robotics plays an essential role in achieving truly co-creative cyber physical systems. In robotic manipulation tasks, one of the biggest challenges is to estimate the pose of given workpieces. Even though the recent deep-learning-based models show promising results, they require an immense dataset for training. In this paper, two vision-based, multi-object grasp pose estimation models (MOGPE), the MOGPE Real-Time and the MOGPE High-Precision are proposed. Furthermore, a sim2real method based on domain randomization to diminish the reality gap and overcome the data shortage. Our methods yielded an 80% and a 96.67% success rate in a real-world robotic pick-and-place experiment, with the MOGPE Real-Time and the MOGPE High-Precision model respectively. Our framework provides an industrial tool for fast data generation and model training and requires minimal domain-specific data.

Keywords: adaptive robotics, robot vision, sim2real knowledge transfer, smart manufacturing, cyber physical production systems.

1. INTRODUCTION

Adaptive robotics aims to solve challenges arising from the concept of co-creative cyber physical systems. Traditional robotic applications can move objects or assemble parts fast and reliably in a fully-controlled environment that is well-suited for mass production. Applying these traditional applications is not economically feasible for lower volume production such as manufacturing customized products. Additionally, in many situations, robots need to work either physically alongside human workers (human-robot collaboration) or in a workflow where their input is significantly influenced by the dexterity of human workers (Tian et al., 2021; Zhou et al., 2019).

Adaptive robots need to sense and interpret their environment and make informed and automatic decisions on how they maximize their targets. Similarly to humans, vision is an efficient way to perceive the environment.

Even though deep-learning-based models revolutionized the field of computer vision, their applications in the field of robotics have obstacles. Collecting the datasets for these notoriously data-hungry learning-based models, on many occasions, is not feasible in the industry. Levine et al. (2016) recorded over 1.7 million robotic grasp attempts over several months using between 6 and 14 robots at the same time.

Transfer learning(Weiss et al., 2016) in the case of supervised learning can be described as creating synthetic data to train machine learning models. Thus, the tedious work of collecting and labeling data can be omitted. The model trained on synthetic data, ceteris paribus, will not generalize well for the real domain. This is called the reality gap that transfer learning aims to diminish. These methods can generally be grouped into the field of domain randomization and domain adaptation. The former methods try to diminish the reality gap by introducing artificial noise and randomness, thus the model will not overfit on the domain-specific characteristics but learn the underlying data representation of the objects. Whereas the latter approaches attempt to transform the source domain to the target domain (generating photo-realistic images as an example) or transfer the source and the target domain to a third domain.

Robotic grasping is an unsolved problem and a critical challenge of adaptive robotics in which the model not only needs to identify and locate the different parts but estimate its orientation to compute a viable grasp position. The contributions of the paper are as follows:

- The proposed multi-object grasp pose estimation methods (MOGPE), the MOGPE Real-Time and MOGPE High-Precision models.
- The synthetic data generation process with sim2real domain randomization for grasp pose estimation.
- Our freely available implementation of the grasping pose estimation [1] and the robot control framework [2].

[1] https://git.sztaki.hu/emi/grasping-pose-estimation

[2] https://git.sztaki.hu/emi/robot_control_framework

10.1016/j.ifacol.2023.10.121

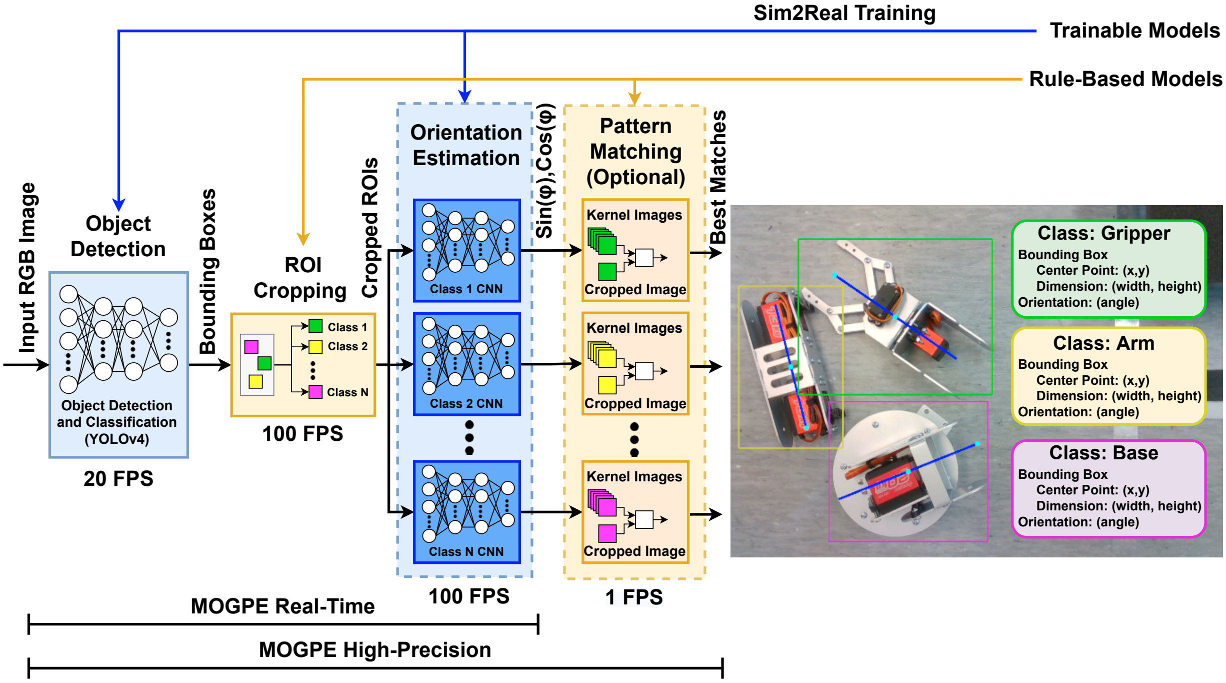


Fig. 1. Illustration of our multi-object grasp pose estimation method.

Our results:

- In our case study, the object detection model yielded a 98.78% mAP$_{50}$ score, while the orientation estimation models achieved a 97.04% success rate on average.
- The MOGPE Real-Time (RT) model runs in real time. The object detection stage works at 20 FPS while the orientation estimation stage runs at 100 FPS.
- In a real-world experiment of robotic grasping, the MOGPE RT model achieved an 80% while the MOGPE High-Precision (HP) model accomplished a 96.67% success rate. These results serve as a proof-of-concept of our approach.

This work is a continuation of our previous work (Horváth et al., 2022) where a sim2real framework for object detection was proposed.

2. PROBLEM STATEMENT

In this section, the problem is briefly presented alongside our approach. For a complete overview of the field the reader is refered to survey articles such as Kleeberger et al. (2020)

The problem defined as a 3.5 DoF (x,y,θ,c) pick-and-place robot manipulation task. The planar coordinates (x,y), the θ angle of the orientation and the classes (c) of the objects need to be estimated. Further characteristics of the problem are as follows. The position of the plane where the objects are placed must be known. The objects are recognized only from one of their stable position. The parts are separated and all object classes are present at the training.

The given model needs to identify and locate all the different workpieces and then estimate the orientations of them. Additional challenges arise from the following circumstances. The environment is not controlled (no special illumination), and the background is not simplified (no monochromatic background). The model has access to only one RGB image, thus the 3D reconstruction of the scene is not possible. The grasp must be performed with a two-finger gripper and every object has only one grasp position.

Our solution is a two-stage, data-driven (supervised learning), still 3D model-based method. As our aim is industrial usability, the assumption is that the availability of real-world data is limited. The majority of the training dataset is synthetic, generated by our sim2real domain randomization framework.

3. RELATED WORKS

The related works focus mostly on two aspects of the robotic grasping challenge:

(1) What is the optimal model to solve the problem?
(2) How to generate training data and then transfer the knowledge to the real world?

Mahler et al. (2019) introduced Dex-Net 4.0. They use a simulator to create a training dataset for their Grasp Quality Convolutional Neural Network. Even though this approach is relatively strong in bin-picking tasks, it is less optimal for pick-and-place operations with predefined grasping positions.

Tobin et al. (2018) propose an autoregressive grasp planning that gives a probability distribution over possible grasps. They used the YCB (Calli et al., 2015) dataset and in a real-world scenario, they achieved an 80% success rate.

Pashevich et al. (2019) trained a model to learn manipulation policies in a simulation using depth images and sim2real transfer. They achieved 1.09 ± 0.73 cm positional error in the real world. Furthermore, in the tasks of cube picking, cube stacking, and cube placing tasks, they yielded 19, 18, and 15 successful attempts out of 20.

Zhang et al. (2019a). presented how to efficiently transfer visuo-motor policies from simulation to real-world. In their case study, a velocity-controlled 7 DoF robot arm needed to reach a blue cuboid object in a table-top scenario. They achieved a 97.8% success rate and 1.8 cm control accuracy.

Zhang et al. (2019b) introduced a two-stage ROI-based robotic grasp detection model focusing object overlapping scenes. They yielded 92.5% and 83.8% success rate, respectively in single-object and multi-object scenes. Nevertheless, using real images, they did not focus on sim2real knowledge transfer.

It is challenging to compare the works above as many aspects of the problem are different. However, in general, 80% success rate is considered a good performance. In our case, industrial usability is an important factor, thus our aim is to reach close to 100% success rate keeping universality as much as possible.

It is important to mention that, according to our best knowledge, even though there are existing industrial solutions for some types of robotic grasping, they cannot perform the task described in Section 2. In general, these tools either detect a tag on a palette and then move to predefined positions on the palette, or they are only capable of detecting one class of objects, or they use many real-world images. For the aforementioned reasons, the comparison of such solutions is not feasible.

4. APPROACH

In our approach, the problem is divided into two stages. In the first stage, the different objects are localized (bounding box information with classification). In the second stage, the orientations of the detected objects are estimated with convolutional neural networks trained on class-specific examples. As the plane coordinates and the 3D models of the objects are known, with the center points and the orientations, the grasping position can be calculated effortlessly. The illustration of the proposed approach is shown in Fig. 1.

In Section 4.1 the object detection model is presented, while in Section 4.3 the orientation estimation is described. These two stages are the main building blocks of the MOGPE RT model. In Section 4.2, the region of interest (ROI) cropping algorithm is presented which connects the two stages of the model. In Section 4.4, the MOGPE HP model is described, which is an extension of the MOGPE RT model. Our implementation is available at[3].

4.1 Object Detection (Stage 1)

The object detection model needs to locate and classify all the objects on an image. In robotics, the object detection model not only needs to be precise (high value of mAP, mean area under the curve) but also needs to work fast (high FPS, frame per second). The object detection stage is built on our previous work (Horváth et al., 2022). For the convolutional neural network (CNN), YOLOv4 (Bochkovskiy et al., 2020) was chosen as it has an optimal accuracy-speed trade-off.

The network is trained on domain randomized synthetic images combined with one real example[4]. Instead of sequentially training the model on the source domain (synthetic images) and then fine-tuning it on the target domain (real images), the model was trained in parallel. The real data was multiplied to have equal weight in the training process to the generated synthetic data. Thus, the model learns the domain shift and the generalization simultaneously

For the synthetic data generation, the 3D models of the objects are loaded into the simulator with randomized positions and randomized textures. The camera renders images from randomized positions and the labels are generated automatically, knowing the position of the objects and the camera. Furthermore, a post-processing method is executed to introduce additional artificial noise. For further details of the sim2real object detection framework, and the synthetic data generation process, the reader is referred to Horváth et al. (2022).

With this method, the reality gap could be shrunk to a satisfactory level, meaning that the model is capable of accurately locating and classifying the different objects not only in simulation but in the real world as well. The data generation lasts around 0.25 - 0.5s per image, while the training takes 12h on a GeForce RTX 2080 Ti GPU. The model prediction time is above 20 FPS on a GeForce RTX 3060 GPU.

4.2 ROI Cropping

Between the first and second stages, a rule-based algorithm cuts out the specific ROIs of the objects from the input image according to the bounding box information. Then, it transforms them to the appropriate size while keeping the orientation of the objects (one object per image) and forwards them to the specific CNN of the second stage, depicted in Fig. 1 and detailed in Fig. 2. Assuming that there are N classes, an object that is detected on the image can be sent to N different CNNs.

As the next stage must estimate the orientation of the objects, it is crucial that the image transformation does not change the orientation. For this reason, the image is padded with zeros to a square and then resized to the expected input size of the neural network. In our case, it is 300x300.

4.3 Orientation Estimation (Stage 2)

The second stage of the model is the orientation estimation which contains N CNNs (each for one class). Each of them takes a 300x300x3 image as input and outputs the sine and cosine representations of the orientation. Learning the sine and cosine values instead of learning purely the angles was

[3] https://git.sztaki.hu/emi/grasping-pose-estimation

[4] Data augmentation was applied according to Bochkovskiy et al. (2020)

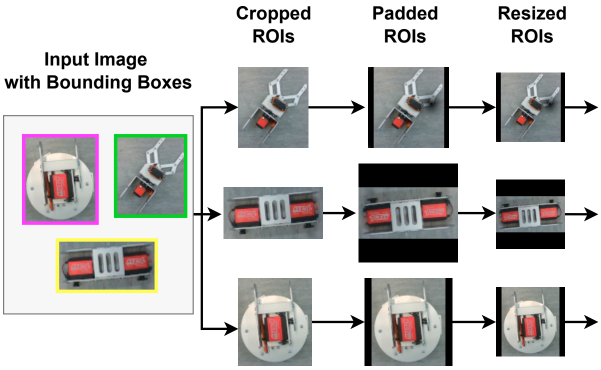


Fig. 2. The data flow of the ROI cropping method

chosen as the former method is a better fit for regression problems as in these trigonometric functions the distances between angles next to each other are continuous. Having computed these values, the orientation can be calculated using the `atan2` function.

The architecture of the CNNs is shown in Fig.3. In the feature extractor, there are 4 convolutional layers with ReLU (rectified linear unit) activation functions and each of them is followed by a MaxPooling layer. To compute the outputs, there are 4 fully connected layers in the head of the network. The models are trained from scratch, independently from each other, on class-specific synthetic and real examples.

The synthetic data were generated in PyBullet. The 3D model of the object is placed in the simulator and rotated around the z-axis (perpendicular to the plane where the object is placed) while random textures are added to the plane and to the object as well. For each bit of rotation, an image is taken and the label is automatically generated with it. Some examples can be seen in Fig. 4. The data generation lasts around 0.25 - 0.5s per image, while the training, implemented in PyTorch, takes 2.5h per class on a GeForce RTX 2080 Ti GPU. The model prediction time is at 100 FPS on a GeForce RTX 3060 GPU.

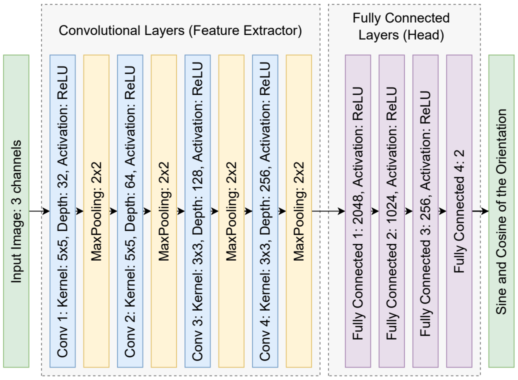

Convolutional Layers (Feature Extractor)
Fully Connected
Layers (Head)
Input Image: 3 channels
Conv 1: Kernel: 5x5, Depth: 32, Activation: ReLU
MaxPooling: 2x2
Conv 2: Kernel: 5x5, Depth: 64, Activation: ReLU
MaxPooling: 2x2
Conv 3: Kernel: 3x3, Depth: 128, Activation: ReLU
MaxPooling: 2x2
Conv 4: Kernel: 3x3, Depth: 256, Activation: ReLU
MaxPooling: 2x2
Fully Connected 1: 2048, Activation: ReLU
Fully Connected 2: 1024, Activation: ReLU
Fully Connected 3: 256, Activation: ReLU
Fully Connected 4: 2
Sine and Cosine of the Orientation

Fig. 3. The proposed CNN architecture for orientation estimation

4.4 Pattern Matching

In the industry, it is essential that the robot grasps an object successfully and with precision. For this reason, a rule-based pattern matching algorithm is executed after the orientation estimation. The pattern matching is performed locally, in the neighborhood of the estimated orientation. With this addition, the model achieves higher precision at the cost of the extra computation. The pattern matching algorithm is the difference between the two proposed models, only the MOGPE High-Precision model incorporates this step.

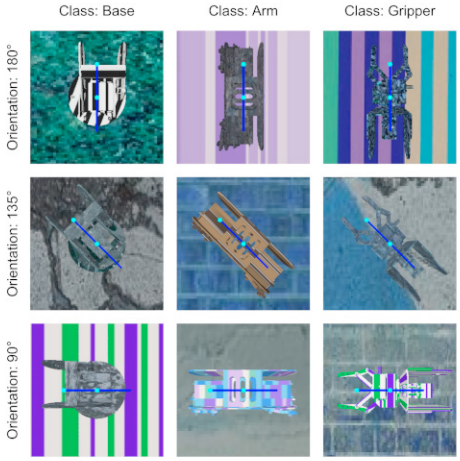


Fig. 4. Some examples of the generated synthetic training dataset

The pattern matching algorithm compares the image of the object with a set of precomputed rotated kernel images. For one class, one real kernel image is rotated 359 times making 360 rotated kernel images [5] .

Comparing two images takes around 13 ms, thus if the search is restricted for ± 10 degrees with a 1-degree resolution, it takes 0.26 seconds. While performing it in the whole range (without the orientation estimation by the CNN) takes 4.68 seconds.

It is important to note, that the pattern matching algorithm needs a good initialization, provided by the orientation estimation CNN. Otherwise, it frequently finds wrong orientations, especially in symmetric objects.

5. ROBOT CONTROL ARCHITECTURE

In this section, the robot control architecture is presented which shows how our computer vision models can be utilized in real-world robotic applications.

The robot control architecture is based on ROS (robot operating system) and is depicted in Fig.5. The `camera driver` node publishes the images that are first read by the `object detection` node which then publishes the bounding box information. Based on these, the `orientation estimation` node predicts the orientation of the visible objects, and sends this information to the `pattern matching` node which returns with the corrected orientation estimate when the `get orientation service` is called. In case of the MOGPE RT model, the `get orientation service` returns with the original value of the `orientation estimation` node. The `camera frame`

[5] If the precision needs to be higher than 1 degree, this procedure can be done on a finer scale.

broadcaster node publishes the transformation between the camera frame and the end effector. With this information, the pixel converter node transforms pixel coordinates to the word frame when the convert point service is called. For motion planning, the MoveIt framework (Görner et al., 2019) was used. Our implementation is available at [6]

The camera is calibrated using the VISP library (Marchand et al., 2005). By taking some pictures of a known pattern (a chessboard in our case), the transformation between the robot's end effector frame and the camera frame is calculated [7] . Since the position of the plane where the objects are placed and the 3D models of the objects are known, the inverse perspective projection equations can be used to transform object positions from the image frame to the camera frame, then transform them to the world frame using the transformation matrix obtained from the calibration.

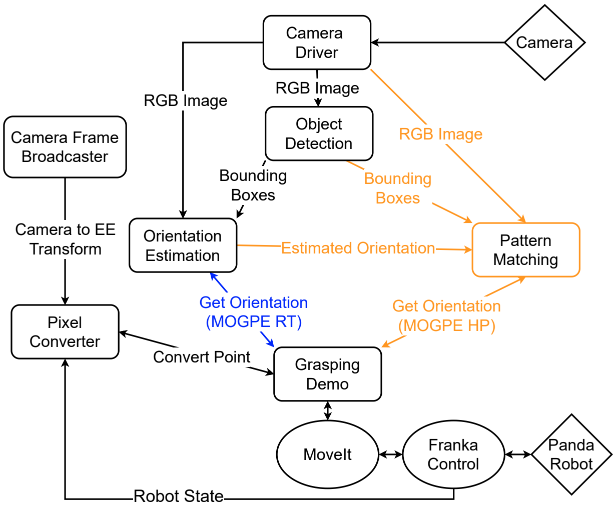


Fig. 5. The robot control architecture. With blue color, the version of MOGPE RT model, while with orange color, the version of MOGPE HP model.

6. RESULTS

In this section, the evaluate of our approach is presented. First, the settings of the experiment are described in Section 6.1. Then, the results of the object detection (Section 6.2), the orientation estimation models (Section 6.3), and the real-world robotic grasping (Section 6.4) are presented.

6.1 Setting of the Robotic Experiments

For this robotic case study, three industrial parts were selected that are themselves parts of a simple robot arm, shown in Fig. 6. Synthetic samples of the parts are depicted in Fig. 4.

Initially, the parts are randomly placed in the starting area. The task of the robot is to pick and place the parts one by one from the starting area to the designated target positions using its two-finger gripper. Neither special illumination was applied nor monochromatic background.

Table 1. The mAP_{50} scores of the object detection model

	Dataset		
	Train	Valid	Test
Training #1	100%	100%	98.85%
Training #2	100%	100%	98.81%
Training #3	100%	99.81%	98.85%
Training #4	100%	100%	98.81%
Training #5	97.07%	95.26%	98.56%
AVG.	**99.41%**	**99.01%**	**98.78%**
STD.	**1.3103%**	**2.1001%**	**0.1224%**

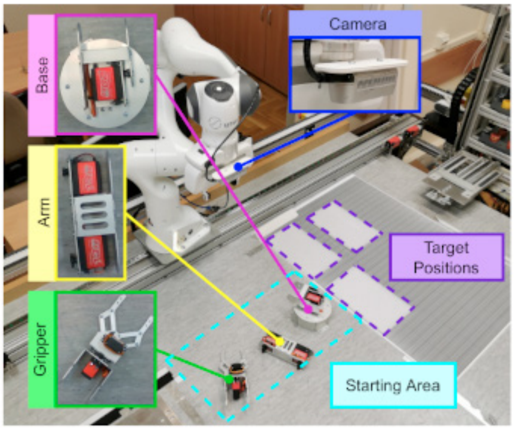


Fig. 6. Experimental setup.

6.2 Object Detection

To train the model [8], 2000 synthetic images were generated alongside one real image (with all the 3 objects) multiplied 2000 times. The batch size was set to 64, and the other hyper-parameters of the training were chosen according to the recommendation of Bochkovskiy et al. (2020).

The quantitative evaluation is shown in Tab. 1. The validation dataset was generated from the same distribution as the training dataset. On the other hand, the test dataset contains 59 real images, taken in different environmental and illumination conditions. Achieving 98.78% mAP_{50} on average can be considered a robust performance as it is close to the performance achieved on the training (99.41% mAP_{50}) and on the validation datasets (99.01% mAP_{50}). Having a reliable output of the object detection stage is crucial as this output is the input of the orientation estimation. As it is shown in our previous work (Horváth et al., 2022), the object detection part works for more classes as well, and as in the orientation estimation stage, every class processed separately, our method can be easily scaled up to more classes.

For qualitative evaluation, Fig. 7 shows two accurate examples of object detection. More qualitative evaluation can be found at [9] .

[6] https://git.sztaki.hu/emi/robot_control_framework

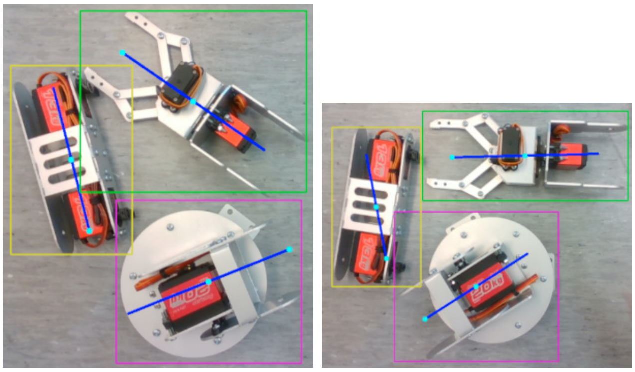
[7] https://visp-doc.inria.fr/doxygen/visp-daily/tutorial-calibration-extrinsic.html

[8] Pre-trained on ImageNet

[9] https://youtu.be/luwA6RDEaoA

Table 2. The success rate of the pose estimation model. A successful estimation is defined as within 10 degrees to the ground truth. 'Train S.' and 'Train R.' are abbreviations for the training datasets of synthetic and real images.

Dataset	Base	Arm	Gripper	**AVG.**	**STD.**
Train S.	99.76%	98.71%	99.54%	**99.34%**	**0.55%**
Train R.	99.85%	99.75%	99.83%	**99.81%**	**0.05%**
Valid	100.0%	99.16%	99.72%	**99.63%**	**0.42%**
Test	99.17%	92.22%	99.72%	**97.04%**	**4.18%**



(a) Accurate example (b) Inaccurate example

Fig. 7. An accurate (a) and an inaccurate (b) prediction. The orientation of the arm is slightly tilted in the latter case. Regarding the object detection, both examples are accurate.

6.3 *Orientation Estimation*

To train the orientation estimation CNNs, 4320 synthetic annotated images were generated. As a real dataset 15, 12, and 12 real images were available from the classes of the base, arm, and gripper. These real images were also augmented by rotating them 359 times which resulted in (with the original one) 5400, 4320, and 4320 images per class. 720 (2 times 360) real images were taken away per class for validation and testing. The loss function is MSE with Adam optimizer and the learning rate is 0.001. The batch size is 128, the training time is 100 epoch with early stopping. The loss function converged rapidly both on the training and on the validation set.

For quantitative evaluation, Tab. 2 shows the success rate of the models. In the case of the base and gripper objects, the success rate is above 99% in all datasets. In the case of the arm, the model achieves a 92% success rate on the test dataset. The main reason behind this phenomenon is the fact that the arm object is more symmetric than the other objects. Thus, shrinking the range of estimation to 180 degrees would increase the performance.

For qualitative evaluation, Fig. 7 shows an accurate and an inaccurate example of orientation estimation. More qualitative evaluation can be found at[10].

6.4 *Robotic Grasping*

Finally, the performance of our models was measured in a real-world robotic grasping experiment using a 7 DoF collaborative robot. Ten grasp attempts were made per class and per model (all in all 60 grasp attempts). The results of the experiments are summarized in Tab. 3. The MOGPE Real-Time model worked well in the case of the arm and gripper classes. Nevertheless, it failed to reliably grasp the base class. On the other hand, the MOGPE High-Precision model could successfully grasp the objects most of the times, yielding a 96.67 % success rate. Six grasp attempts are shown at[11].

Table 3. Results of the robotic grasping experiment

Model	Base	Arm	Gripper	**Success rate**
MOGPE RT	5/10	9/10	10/10	**80%**
MOGPE HP	10/10	10/10	9/10	**96.67%**

7. CONCLUSIONS AND FUTURE WORK

In this paper, robotic grasping was addressed, a critical challenge of adaptive robotics which plays an essential role in achieving truly co-creative cyber physical systems. Two vision-based, multi-object grasp pose estimation models were presented, the MOGPE Real-Time and the MOGPE High-Precision. Furthermore, a sim2real knowledge transfer method based on domain randomization to diminish the reality gap and to overcome the data shortage.

Our framework provides an industrial tool for fast data generation and model training and requires minimal domain-specific data. In test time, the model does not only work fast (object detection 20 FPS, orientation estimation 100 FPS) but performs well (98.78% mAP_{50} score, and 97.04% success rate).

Our approach is validated not only on images but in a real-world robotic grasping experiment where the MOGPE RT model achieved an 80%, while the MOGPE HP model accomplished a 96.67% success rate.

In the future, our target is to further improve our sim2real transfer learning methods expecting to gain performance with a more versatile synthetic dataset. Additionally, an interesting continuation would be to experiment with adversarial training and other industrial setups.

ACKNOWLEDGEMENTS

This research has been supported by the ED_18-2-2018-0006 grant on an “Research on prime exploitation of the potential provided by the industrial digitalisation”

The research was supported by the European Union within the framework of the National Laboratory for Autonomous Systems. (RRF-2.3.1-21-2022-00002)

REFERENCES

Bochkovskiy, A., Wang, C.Y., and Liao, H.Y.M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. URL `https://arxiv.org/abs/2004.10934`.

Calli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., and Dollar, A.M. (2015). The YCB Object and

[10] See footnote 9.

[11] See footnote 9.

Model Set: Towards Common Benchmarks for Manipulation Research. In *2015 International Conference on Advanced Robotics (ICAR)*, 510–517. doi: 10.1109/ICAR.2015.7251504.

Görner, M., Haschke, R., Ritter, H., and Zhang, J. (2019). MoveIt! Task Constructor for Task-Level Motion Planning. In *2019 International Conference on Robotics and Automation (ICRA)*, 190–196. doi: 10.1109/ICRA.2019.8793898. ISSN: 2577-087X.

Horváth, D., Erdős, G., Istenes, Z., Horváth, T., and Földi, S. (2022). Object Detection Using Sim2Real Domain Randomization for Robotic Applications. *IEEE Transactions on Robotics*, 1–19. doi: 10.1109/TRO.2022.3207619. Early Access.

Kleeberger, K., Bormann, R., Kraus, W., and Huber, M.F. (2020). A Survey on Learning-Based Robotic Grasping. *Current Robotics Reports*, 1(4), 239–249. doi: 10.1007/s43154-020-00021-6.

Levine, S., Pastor, P., Krizhevsky, A., and Quillen, D. (2016). Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. *The International Journal of Robotics Research*. doi:10.1177/0278364917710318.

Mahler, J., Matl, M., Satish, V., Danielczuk, M., DeRose, B., McKinley, S., and Goldberg, K. (2019). Learning Ambidextrous Robot Grasping Policies. *Science Robotics*, 4, eaau4984. doi:10.1126/scirobotics.aau4984.

Marchand, E., Spindler, F., and Chaumette, F. (2005). ViSP for Visual Servoing: A Generic Software Platform with a Wide Class of Robot Control Skills. *IEEE Robotics and Automation Magazine*, 12(4), 40.

Pashevich, A., Strudel, R., Kalevatykh, I., Laptev, I., and Schmid, C. (2019). Learning to Augment Synthetic Images for Sim2Real Policy Transfer. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2651–2657. doi: 10.1109/IROS40897.2019.8967622.

Tian, J., Vanderstraeten, J., Matthyssens, P., and Shen, L. (2021). Developing and Leveraging Platforms in a Traditional Industry: An Orchestration and Co-Creation Perspective. *Industrial Marketing Management*, 92, 14–33. doi:10.1016/j.indmarman.2020.10.007.

Tobin, J., Biewald, L., Duan, R., Andrychowicz, M., Handa, A., Kumar, V., McGrew, B., Ray, A., Schneider, J., Welinder, P., Zaremba, W., and Abbeel, P. (2018). Domain Randomization and Generative Models for Robotic Grasping. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3482–3489. doi:10.1109/IROS.2018.8593933.

Weiss, K., Khoshgoftaar, T.M., and Wang, D. (2016). A Survey of Transfer Learning. *Journal of Big Data*, 3(1), 9. doi:10.1186/s40537-016-0043-6.

Zhang, F., Leitner, J., Ge, Z., Milford, M., and Corke, P. (2019a). Adversarial Discriminative Sim-to-Real Transfer of Visuo-Motor Policies. *The International Journal of Robotics Research*, 38(10-11), 1229–1245. doi: 10.1177/0278364919870227. Publisher: SAGE Publications Ltd STM.

Zhang, H., Lan, X., Bai, S., Zhou, X., Tian, Z., and Zheng, N. (2019b). ROI-based Robotic Grasp Detection for Object Overlapping Scenes. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4768–4775. doi: 10.1109/IROS40897.2019.8967869. ISSN: 2153-0866.

Zhou, J., Zhou, Y., Wang, B., and Zang, J. (2019). Human–Cyber–Physical Systems (HCPSs) in the Context of New-Generation Intelligent Manufacturing. 5(4), 624–636. doi:10.1016/j.eng.2019.07.015.