IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, October 2018.

Robust 6D Object Pose Estimation in Cluttered Scenes using Semantic Segmentation and Pose Regression Networks

Arul Selvam Periyasamy, Max Schwarz, and Sven Behnke

***Abstract*— Object pose estimation is a crucial prerequisite for robots to perform autonomous manipulation in clutter. Real-world bin-picking settings such as warehouses present additional challenges, e.g., new objects are added constantly. Most of the existing object pose estimation methods assume that 3D models of the objects is available beforehand. We present a pipeline that requires minimal human intervention and circumvents the reliance on the availability of 3D models by a fast data acquisition method and a synthetic data generation procedure. This work builds on previous work on semantic segmentation of cluttered bin-picking scenes to isolate individual objects in clutter. An additional network is trained on synthetic scenes to estimate object poses from a cropped object-centered encoding extracted from the segmentation results. The proposed method is evaluated on a synthetic validation dataset and cluttered real-world scenes.**

I. Introduction

Many robotic applications depend on the robust estimation of the object poses. Different tasks may, however, place varying emphasis on certain aspects, such as preciseness, robustness, speed, or fast adaption to new objects. Robustness, in the sense of general applicability across different objects, is difficult to obtain: Transparent or featureless objects pose challenges for many pose estimation methods that require valid depth measurements and/or rely on a fused 3D model.

Our work is motivated by the Amazon Robotics Challenge 2017, which required participants to manipulate objects in cluttered bin picking scenes. In particular, fast and robust adaption to novel items was required, with teams only having 30 minutes to enroll 16 new objects to their systems. In this context, leveraging high-quality object models is difficult.

We present a Convolutional Neural Network (CNN)-based regression pipeline for 6D pose estimation from RGB-D image segments, which can be used following a semantic segmentation of the scene (see Fig. 1). It first estimates a 5D pose from the RGB image, whose translation is then projected into 3D using the depth information.

Our contributions include

1) a fast object enrollment scheme for quickly learning new items from automated turntable captures,
2) an encoding mechanism for focusing the network on a particular object in a cluttered scene,
3) a CNN pipeline for direct pose regression, and
4) explicit handling of object symmetries.

All authors are with Autonomous Intelligent Systems, University of Bonn, Germany. periyasa@ais.uni-bonn.de

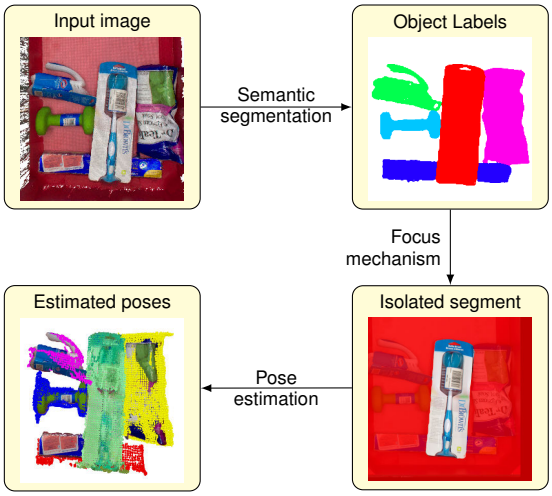

Fig. 1. The proposed 6D pose estimation pipeline for cluttered scenes.

II. Related Work

Traditionally, pose estimation is done using registration-based methods. Given a good 3D model of the object and a clear separation from the background, Iterative Closest Point algorithms and their extensions perform extremely well [1]. In cases where depth measurements are unreliable or a 3D object model is hard to obtain, more complex methods are required.

RGB-only methods perform more robustly in these settings. For example, Zhu *et al.* [2] extracted object silhouettes and matched these against 3D models.

Hybrid learning-based methods used both RGB and depth information, adapting to impreciseness or missing depth information. Schwarz *et al.* [3] demonstrated object detection and pose estimation using pretrained CNN features. However, only 1D pose (yaw angle) was estimated.

Krull *et al.* [4] used random forests to generate an object probability distribution as in semantic segmentation and 3D object coordinates corresponding to each object for each pixel in the image. Then for each detected object, in a Random Sample Consensus (RANSAC)-like approach, 3-tuples of pixels were randomly selected and the affine transformation to their predicted 3D object coordinates was estimated. This produced a set of hypotheses H using the Metropolis algorithm. RGB and depth images were rendered from the 3D model for each of the hypotheses. The authors then trained a CNN to take the rendered and observed images as input and output an energy value that is high if inputs are similar and low when dissimilar. Finally, the hypothesis

with maximum energy was selected. In contrast, our work needs one forward pass per segmented object, but cannot take advantage of iterative refinement.

Koo *et al.* [5] trained a CNN to regress the pose of parts using multiple rendered views of CAD models. The method needed depth information to pick a single part from a pile. We alleviate the need for depth information to isolate an object in the clutter by computing a semantic segmentation of the scene and performing the pose estimation only on a crop of the scene.

Wohlhart and Lepetit [6] trained a CNN not to estimate the pose of the objects directly but to compute a feature descriptor such that the Euclidean distance between the descriptors corresponding to the two poses are smaller if the poses are similar and larger when the poses are dissimilar.

Kendall *et al.* [7] and Kendall and Cipolla [8] used CNNs to regress the 6D camera pose from a single RGB image in a large-scale outdoor setting. Note that this inverse problem does not require an attention mechanism to focus the estimation on a particular object. Both Koo *et al.* [5] and Kendall *et al.* [7] encoded orientation as a quaternion as we do in our approach.

In contrast to other RGB-D methods, our method uses depth only during the capturing process before training the model, and for projecting the inferred 5D pose estimate to 6D. This increases the robustness against missing depth and allows 5D pose estimation on RGB-only images.

There are multiple publicly available datasets for benchmarking object pose estimation methods, Some include Washington RGB-D [9], SUN RGB-D [10], and OccludedLinemod [11]. However, these are often focused on a small set of objects, limiting generalizability. In recent works, such as SceneNet RGB-D [12], we can see a trend towards large-scale datasets covering all kinds of objects, similar to ImageNet [13] in the image classification domain. In contrast, our work focuses on a particular robotic application with fast adaption to a novel set of objects.

III. 6D Pose Estimation

We propose a pose estimation network based on direct regression of a 5D pose, i.e. 2D translation in the image plane and 3D orientation. The 2D translation is later projected into 3D using depth information.

A. Data Acquisition & Training Scene Synthesis

For many applications, it is important to quickly adapt to novel object classes. This adaption does not only encompass training the pose regressor, but also capturing the necessary training data. Many works ignore this step and instead require high-fidelity 3D models as training input, which can be hard to acquire.

In order to keep necessary human intervention minimal, we use an automated turntable setup, which we designed for the Amazon Robotics Challenge 2017 (see [14] for details). The turntable shown in Fig. 2 captures 20 views from all directions in 10 s with an Intel RealSense SR-300 RGB-D sensor. We call this set of views *sequence*. For most of

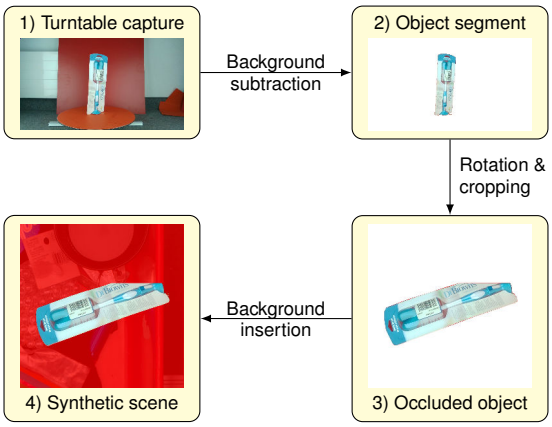

Fig. 2. Data acquisition & scene synthesis pipeline.

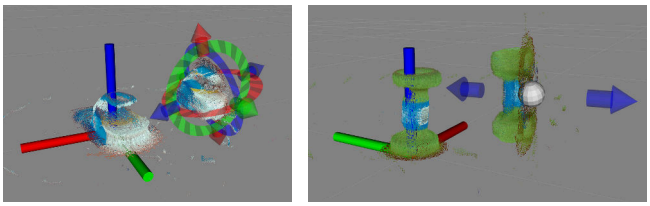

Fig. 3. Alignment tool usage. The coordinate system shows the frame of reference for the particular object. Already aligned sequence point clouds are shown around this origin. A new cloud is integrated by transforming it interactively using the 6D marker.

the objects three sequences are captured in different resting orientations on the turntable, resulting in 60 views per object. The object is segmented in each frame using background subtraction (i.e. comparison with an object-less frame).

The turntable images are not immediately suitable for training, since they show the object in an isolated setting without occlusions. To address this issue, we introduce a scene synthesis step, which overlays the turntable images of objects onto complex scenes. For generating training data for semantic segmentation, we start with images of complex scenes that were manually annotated beforehand. We overlay the turntable images onto it—occluding the already existing objects—and concurrently generate pixel-wise ground truth annotation. For training the pose estimation, we just use the same background images (without annotation). A special encoding scheme, discussed in the next subsection, is used to focus the network on a single object in a complex scene, eliminating the need for background annotation.

The ground truth pose of the objects is generated using forward kinematics from the measured turntable angle. The alignment of poses between different sequences is done manually using an RViz-based[1] GUI (see Fig. 3). Also, the available set of poses is augmented by sampling random rotations around the camera axis.

B. Encoding Of Object In Focus

Since scenes may have multiple objects, we need a mechanism to make the network focus on the object of interest.

[1] `http://wiki.ros.org/rviz`

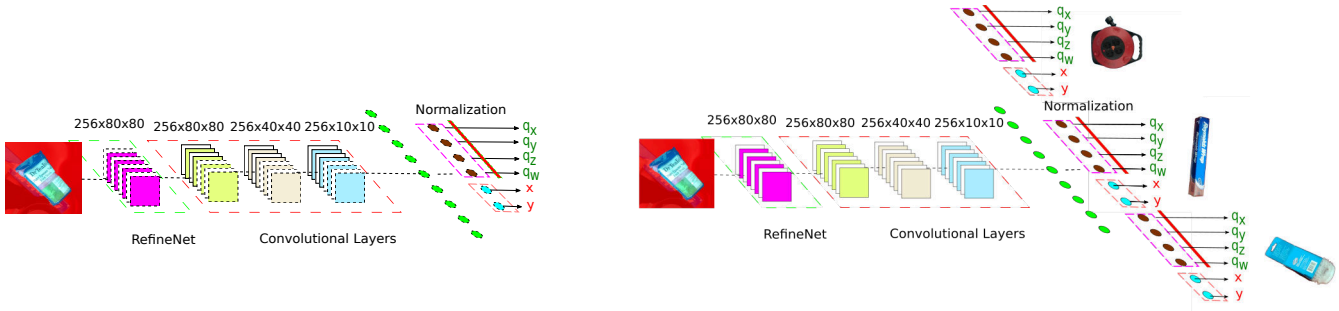

Fig. 4. Pose estimation network architectures. Left: Single-block output variant. Right: Multi-block output variant.

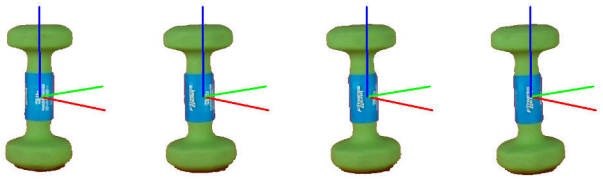

Fig. 5. Resolution of symmetries. Invariant poses are assigned the same ground truth pose.

Fig. 6. ARC objects used in pose estimation evaluation. From left to right, top to bottom: Browns brush, epsom salts, reynolds wrap, hand weight, and utility brush.

We encode the object of interest by pushing other pixels towards red (see Fig. 2 Step: 4)). This encoding is natural in the setting of our test dataset (see Section IV-A), which has red background behind the objects. During training, this encoding is based on the background subtraction mask; during inference, the semantic segmentation prediction is used.

C. CNN Backbone

CNN-based methods can leverage pretraining from large-scale datasets. Since these datasets often aim at the image categorization task (e.g. ImageNet [13]), the spatial resolution of the high-level features extracted by higher levels of these CNNs is quite low. In tasks such as semantic segmentation, this limits performance.

The RefineNet architecture [15] mitigates this problem by successively merging upsampled high-level feature maps with lower-level representations, thus yielding a both highly semantic and spatially precise result. In our work, we use the RefineNet architecture with ResNeXt [16] as the pretrained backbone network. The RefineNet architecture is pretrained on the semantic segmentation task.

D. Semantic Segmentation Network

The semantic segmentation network (see [14] for details) consists of RefineNet backbone followed by one convolutional layer that reduces the number of feature maps to the number of object classes and pixel-wise SoftMax. It is trained to minimize the pixel-wise cross-entropy segmentation loss on the synthetic dataset described in Section III-A.

E. Pose Estimation Network

Our pose estimation network illustrated in Fig. 4 consists of three dimension-reducing convolutional layers of kernel size 3 and stride 2 followed by two fully-connected layers and six output neurons corresponding to the 2D translation in the image plane and the orientation represented as a unit quaternion. All layers use ReLU activations. Finally, an $L2$ normalization layer is added in the orientation part, which guarantees a unit quaternion output, alleviating the need for the network to learn the normalization.

Quaternions are not unique representations of rotation (i.e. q is an equivalent rotation to $-q$). To deal with this non-uniqueness in the representation, we require $q_w \geq 0$ for the ground-truth quaternion during the training phase.

For pose estimation, the scene is cropped to a size of 320×320, centered at the origin of the object in the image plane. Additionally, we randomly move the center of the crop to be a few pixels away from the origin of the object in the image plane. Without this data augmentation, the objects in the training images always appear to be centered in the crop, and the model might learn to overfit this artificial condition. During the inference step, where the ground truth origin of the object is not known, the center of the contour extracted from semantic segmentation is used.

Spatially, the backbone network reduces the 320×320 input scene to 256 feature maps of size 80×80 which are provided as input to the pose estimation network.

The network is trained to minimize the weighted sum of mean-squared-error (MSE) of the translation and the orientation components. The translation error $||\hat{Y}^{xy} - Y^{xy}||_2$ (pixel distance in a 320×320 image) is of a different scale compared to the quaternion distance $||\hat{Y}^{q} - Y^{q}||_2$. We scale the x and y translation to lie in $[-1, 1]$, which brings the translation and orientation errors into the same scale. This allows us to use a simple weighting scheme for the loss components:

$$\text{Loss}(\hat{Y}, Y) = \alpha||\hat{Y}^{xy} - Y^{xy}||_2^2 + (1-\alpha)||\hat{Y}^{q} - Y^{q}||_2^2,$$

where $\hat{Y}_i$ is the ground truth pose, Y_i is the predicted pose for an image i, and Y_i^{xy} and Y_i^{q} are translation and quaternion component, respectively.

We empirically determined a value of 0.7 for α. The network is trained using the Adam optimizer [17].

F. Multi-class Regression

The pose network is trained to estimate the pose of objects belonging to different classes. It is not immediately obvious how to handle this properly. On the one hand, one can require the network to recognize the object class and output the proper pose. A second possibility is to predict *conditional* poses, one for each object class. An external module (for example semantic segmentation) then picks the correct output for the detected class.

For the second case, we implemented a multi-block output variant of the pose estimation network, also shown in Fig. 4. The network has $6N$ outputs for N objects classes. During training, the loss function is only applied to the outputs of the correct object class. Both variants are evaluated in Section IV-B.

G. Symmetries

Some objects may exhibit symmetrical appearance when rotated along a particular axis. For example, the dumbbell object shown in Fig. 5 is symmetrical—with minor differences in the text on the label—to rotation in yaw axis (in blue) and to a rotation of 180° in roll or pitch (red and green). Forcing the network to learn the exact pose from the images that exhibit very little variance may hinder the learning process. Also, learning the pose component that vanishes under (perceived) symmetry is not helpful for robotic manipulation. To deal with the symmetries in the object pose, we assign the same ground truth pose to all poses that are symmetrical. Figure 5 shows an example. The symmetry axes need to be specified manually during alignment.

IV. Evaluation

A. Training Dataset

Our training set consists of five difficult objects from the Amazon Robotics Challenge 2017 (see Fig. 6). We selected objects that require precise grasping because they are heavy or unwieldy, and included deformable and articulated objects. For each object, we capture three sequences of 20 views, as described in Section III-A. For each view, we sample 60 new rotations along the camera axis and thus obtain 3600 training samples per object. We artificially occlude portions of the image to make the network robust against occlusion that might occur in real-world scenarios. The maximum occlusion percentage is limited to 50 %. We split the dataset randomly into training and validation sets with a ratio of 80:20.

TABLE I
Multi- vs. Single-block Output.

			No occlusion	Occlusion
Training	Translation [pix[1]]	Single	9.57	**11.21**
		Multi	**9.28**	12.06
	Orientation [°]	Single	5.92	**6.44**
		Multi	**5.78**	6.56
Validation	Translation [pix[1]]	Single	10.52	**12.14**
		Multi	**9.68**	12.91
	Orientation [°]	Single	7.9	**9.76**
		Multi	**7.4**	9.64

Shown are translation and rotation errors on the validation set.
[1] Relative to the 320×320 input crop centered on the object.

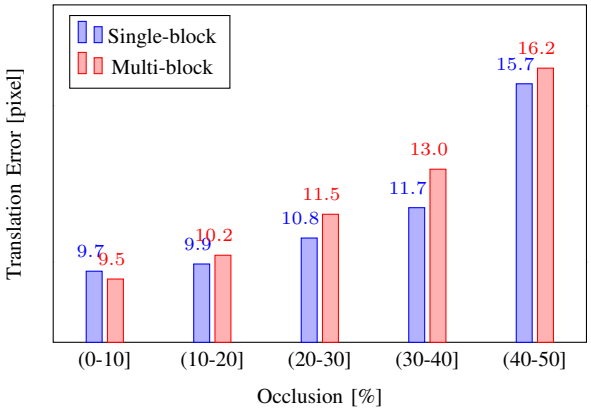

Fig. 7. Effect of occlusion on the translation error. The images in the validation set are grouped into different bins based on the percentage of pixels occluded.

B. Single-block Output vs. Multi-block Output

We evaluated the performance of the single-block output and multi-block output variants of the pose estimation network on the synthetically generated dataset. The results of the comparison are provided in Table I. To understand the strengths and weaknesses of the architectures, we grouped the validation images into bins based on the percentage of pixels being occluded and analyzed the average error made by the variants in different bins. Figure 7 shows the performance of both variants of the network under varying degrees of occlusion. We observed that the multi-block output variant performs a little better in the absence of any occlusion, but the single-block output variant performs slightly better in the presence of occlusion. The former can be explained by the fact that the training objective for the multi-block

Fig. 8. Experiments on complex scenes. Top row: Real cluttered tote scenes. Middle row: Object points extracted using semantic segmentation (shown in real colors) and predicted object poses visualized using object model clouds (in uniform colors). Bottom row: Same visualization after performing ICP refinement.

TABLE II

SINGLE-BLOCK VALIDATION ERROR ON ARC OBJECTS.

Object	Translation [pixel[1]]		Orientation [°]	
	train	val	train	val
Browns brush	10.3	11.4	7.7	10.3
Epsom salts	11.2	12.5	7.4	10.5
Hand weight	9.6	10.4	2.1	2.6
Reynolds wrap	11.6	11.8	6.3	9.8
Utility brush	12.5	13.6	6.9	10.9

[1] Relative to the 320×320 crop centered on the object.

TABLE III

ABLATION STUDY RESULTS ON THE SYNTHETIC DATASET.

Model variant	Translation [pixel]		Orientation [°]	
	train	val	train	val
FC per class	38.9	44.7	37.2	47.2
FC multi-class	46.4	54.4	42.7	51.9
1 Conv with stride 4	36.8	37.4	36.3	44.5
1 Conv with stride 8	36.4	37.1	25.8	34.2
2 Conv with stride 2	32.3	33.9	11.3	17.0
3 Conv with stride 2	10.2	13.5	4.64	10.8

variant does not penalize for wrong object recognition; we simply discard the poses estimated in the blocks that do not correspond to the objects under consideration, i.e. we do not force the multi-block variant to perform object recognition as a part of pose estimation. On the other hand, the single-block variant may be in less danger of overfitting, resulting in better performance on occluded objects, since it is forced to predict poses for different objects, much like a regularizer introducing noise on the ground truth.

In all remaining experiments, we use the single-output variant. Table II shows detailed quantified results on the validation set.

C. Architecture Ablation Study.

Our design of the pose estimation network architecture is motived by the need to run the pose estimation after a semantic segmentation network using the same backbone network. Thus, we wanted the pose estimation architecture to be as light as possible.

We created individual linear models for each object class separately which we considered as the baseline to evaluate the performance of different architectural designs. The linear model consisted of just one fully connected layer on top the 80×80×256 feature maps to regress the 5D pose. The results presented in the first row of Table III indicate that this model cannot learn the task.

In a multi-class setting, the fully connected model performs even worse (see Table III). Adding more fully connected hidden layers is not feasible due to GPU memory constraints caused by the large size of the input feature maps, which limits the number of hidden neurons. Thus, the architectures that do not reduce the input feature dimension failed to attain adequate performance. This leaves us with two direct options to reduce the size of input dimension: using pooling layers, or using convolutional layers with stride > 1. Here, we use convolutional layers with 256 features and stride > 1. Results from an evaluation of different architectures are presented in Table III. We identified the best architecture to be three dimension-reducing convolutional layers of stride 2. From our perspective, the results indicate that a certain capacity of the model is needed to learn the

TABLE IV

RESULTS ON COMPLEX SCENES.

Object	Translation [pixel]		Orientation [°]	
	Predicted	Refined	Predicted	Refined
Browns brush	12.75	12.14	14.63	14.26
Epsom salts	14.43	16.12	16.45	15.27
Hand weight	16.18	15.42	10.34	9.73
Reynolds wrap	12.78	12.48	18.37	16.04
Utility brush	16.74	15.87	15.75	14.88

TABLE V

GENERALIZATION ERROR.

	Translation [pixel]	Rotation [°]
No occlusion	36.34	33.60
Occlusion	39.52	38.21

pose estimation tasks and that convolutional architectures are well-suited for this purpose.

D. Complex Scenes

We collected 14 real scenes as they might have occurred during the Amazon Robotics Challenge 2017[2]. They contain the five objects in varying levels of occlusion and were manually annotated with ground truth poses. Examples of the scenes are shown in the top row of Fig. 8. The five objects possess different physical properties ranging from shiny surfaces in the case of the reynolds wrap to a symmetrical dumbbell and a deformable salt bag. The predicted 6D poses are overlaid with points of uniform color shown in the middle row of Fig. 8. In general, we can observe that the 6D pose predicted by our method is acceptable. The ability of the model to handle occlusion and undersegmentation is demonstrated in the third column where the objects are lying on top of the other objects and the last column where a portion of the reynolds wrap object is undersegmentated. Using Iterative Closest Point (ICP) refinement, the result can be further improved. The final pose after performing Generalized-ICP (GICP) [18] is shown in the bottom row of Fig. 8. In computing the orientation error for symmetrical objects, for example, along yaw axis for the hand weight object, we ignore the error along the symmetrical axis by the following steps:

1) computing the error between the ground truth and the predicted quaternion q_{error},
2) extracting the angular component Ψ of the q_{error} along the axis of the symmetry,
3) creating a new quaternion representing the rotation by Ψ along the symmetrical axis q', and
4) premultiplying q_{error} with the computed q' and thus rotating out the error along the symmetrical axis.

E. Generalization

The objects that we encounter in the real world are often instances of some object category. Humans do not need to learn to recognize/operate each single instance of the same category, instead transferring the knowledge acquired for some instances to others of the same category. With enough training data, CNNs have been shown to generalize to new instances of same category. In this experiment, we evaluate the generalization properties of the pose estimation networks. We trained the pose estimation networks on a set of four drills, evaluating the performance on a different drill (see Fig. 9). The accuracy of the pose estimation network in predicting the pose of the unseen drill with and without occlusion is shown in Table V. We investigated the pose estimation for the unseen cases further by retrieving the image in the training set with the closest orientation to the predicted one. This allows direct visual comparison of the predicted poses (see Fig. 10 for typical working and failure cases). In the last image image of Fig. 10, while the predicted orientation is comparatively good, the translation prediction is off by a wider margin due the bottom portion of the drill being occluded.

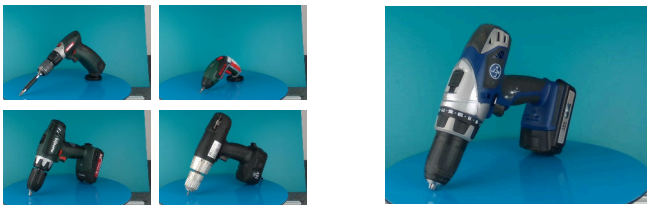

Fig. 9. Objects for the generalization experiment. Left: Training objects. Right: Test object.

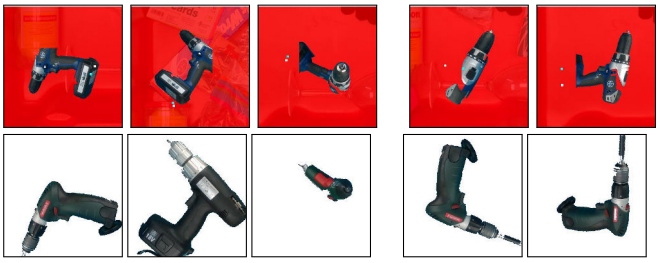

Fig. 10. Generalization with occlusion; Top row: Input image to the network. White and green dots show ground truth and predicted object centers, respectively. Bottom row: Image in the dataset closest to the orientation predicted. The first three images show working cases, while the last two show typical failure cases in orientation and translation, respectively.

F. Symmetries

We quantified the benefits of addressing the symmetry in the pose of the dumbbell by analyzing the loss during the training process as shown in Fig. 11. In the case of not handling symmetries, the model needs around 300 epochs to achieve a reasonable performance compared to the final convergence whereas in the case of handling the symmetries, it needs only around 100 epochs. Thus, the speed of convergence is significantly faster when the symmetry in the poses is handled. The network also converged to a better accuracy. This demonstrates the advantages of dealing with the symmetry in the poses.

[2]The complex scenes dataset is publicly available at http://www.ais.uni-bonn.de/data/pose_estimation/

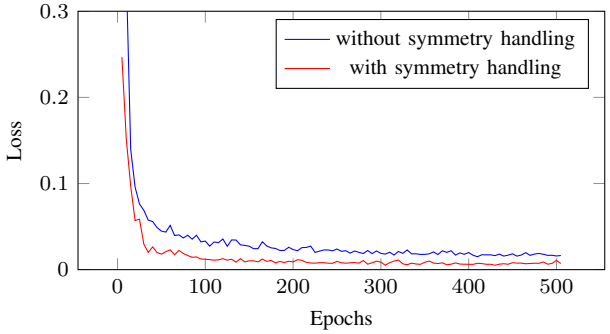

Fig. 11. Effect of symmetry handling on the learning curve.

G. Limitations

One major weakness of our automatic data acquisition pipeline are deformable or articulated objects. Our data capture setup is optimized for fast data acquisition and needs only two or three sequences of turntable captures per object. We apply randomly sampled rotations around the camera axis to synthesize object appearances in new poses. But in the case of deformable or articulated objects, appearance will be affected by gravity or collisions, which is not captured by our data augmentation step. Also, our method relies heavily on the performance of the semantic segmentation module. While undersegmentation is to some degree is modeled as occlusion during training, oversegmentation can affect the performance of pose estimation.

V. Conclusion

We presented a pipeline for 6D object pose estimation in clutter, designed for scenarios where the new objects are to be learned quickly. Our pipeline consists of

1) a fast data acquisition and synthetic training data generation module that needs minimal human intervention,
2) a semantic segmentation module to segment the cluttered scene, and
3) an object pose estimation module that regresses the 5D pose of the objects from the segmented RGB crop of the scene.

We presented and compared two CNN architectures for pose estimation from RGB images and analyzed their strengths and weaknesses in their ability to deal with occlusion. Our method performs semantic segmentation of the scene to extract the crops of objects and estimates the 5D pose of the object: orientation and translation in 2D image plane. The 2D translation prediction in the image plane is projected into 3D using depth information. We also proposed a method to deal with symmetries—similarities on the object appearance under different poses—by assigning the same ground-truth pose for all the poses that exhibit minimal variation in their appearance. Finally, we demonstrated the usability of our method on complex bin-picking scenarios and discussed the performance and limitations of our method.

References

[1] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke, "Registration with the point cloud library: A modular framework for aligning in 3-D," *IEEE Robotics & Automation Magazine*, vol. 22, no. 4, pp. 110–124, 2015.

[2] M. Zhu, K. G. Derpanis, Y. Yang, S. Brahmbhatt, M. Zhang, C. Phillips, M. Lecce, and K. Daniilidis, "Single image 3D object detection and pose estimation for grasping," in *International Conference on Robotics and Automation (ICRA)*, 2014, pp. 3936–3943.

[3] M. Schwarz, H. Schulz, and S. Behnke, "RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features," in *International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1329–1335.

[4] A. Krull, E. Brachmann, F. Michel, M. Ying Yang, S. Gumhold, and C. Rother, "Learning analysis-by-synthesis for 6D pose estimation in RGB-D images," in *International Conference on Computer Vision (ICCV)*, 2015, pp. 954–962.

[5] S. Koo, G. Ficht, G. M. Garcıa, D. Pavlichenko, M. Raak, and S. Behnke, "Robolink feeder: Reconfigurable bin-picking and feeding with a lightweight cable-driven manipulator," in *International Conference on Automation Science and Engineering (CASE)*, 2017.

[6] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3D pose estimation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3109–3118.

[7] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A convolutional network for real-time 6-DOF camera relocalization," in *International Conference on Computer Vision (ICCV)*, 2015, pp. 2938–2946.

[8] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[9] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1817–1824.

[10] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 567–576.

[11] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6D object pose estimation using 3D object coordinates," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 536–551.

[12] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "SceneNet RGB-D: Can 5M synthetic images beat generic imagenet pre-training on indoor segmentation," in *International Conference on Computer Vision (ICCV)*, 2017.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255.

[14] M. Schwarz, C. Lenz, G. M. Garcia, S. Koo, A. S. Periyasamy, M. Schreiber, and S. Behnke, "Fast object learning and dual-arm coordination for cluttered stowing, picking, and packing," in *International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1329–1335.

[15] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[16] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5987–5995.

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Preprint arXiv:1412.6980*, 2014.

[18] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Robotics: Science and Systems (RSS)*, vol. 2, 2009, p. 435.