Learning Implicit Representations of 3D Object Orientations from RGB

Martin Sundermeyer[1], En Yen Puang[1], Zoltan-Csaba Marton[1], Maximilian Durner[1], Rudolph Triebel[1,2]

Abstract— **This work presents a fast and robust algorithm for object orientation estimation that is solely trained on synthetic views rendered from a 3D model. We introduce a dense encoder-decoder architecture that learns implicit representations of 3D object orientations. Since our training is self-supervised, we avoid the necessity of real, pose-annotated training data. Furthermore, it prevents issues related to ambiguous object views. To encode latent representations that are robust against occlusions, clutter and the differences between synthetic and real data, a new domain randomization strategy is proposed. We motivate our approach by experiments on abstract 2D shapes and evaluate it on the challenging T-LESS dataset. In addition to the results in this paper, we provide a live presentation of the system during the workshop, on a Nvidia Jetson TX2 board.**

I. Introduction

Object pose estimation is essential for autonomous manipulation tasks. In recent literature, there exist a lot of approaches (e.g [1], [2], [3]) with different strengths and weaknesses. The significance of object pose estimation is further underlined by the latest Amazon Robotics/Picking Challenge[1] and SIXD Pose Estimation Challenge[2].

Compared to the current state of the art, our approach is significantly different in that it represents object orientations *implicitly*, i.e. we do not train a mapping from input images to explicit pose labels. Instead, we learn a latent representation of object views to retrieve a set of hypotheses for the object orientation. This has several advantages, as we show in the experiments. First, being independent from a concrete representation within $SO(3)$ allows us to handle ambiguous poses caused by symmetric views, because we avoid one-to-many mappings from images to orientations. Second, our implicit representation is learned such that it is robust against occlusion, background clutter and translational variations. And last but not least, our approach does not require any real annotated training data. Figure 1 depicts how our method fits into an object grasping system.

II. Related Work

Deep Learning has been applied to determine object orientations from RGB. However, such approaches rely on large amounts of manually annotated data, and orientation labels are very difficult and cumbersome to produce.

[1] proposed to estimate orientation by having orientation classification heads in its Single Shot Multibox Detector (SSD) [4] style network. This method runs in very high speed and uses synthetic object views. However, symmetry of each object need to be identified and removed manually before training to assure convergence.

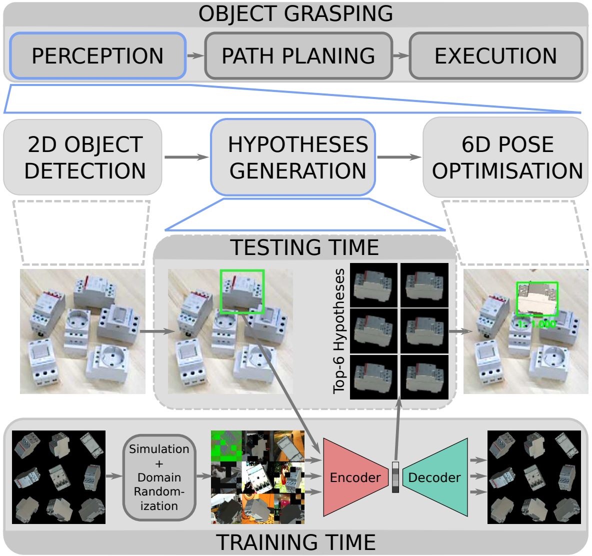

Fig. 1: Our method fits into an object grasping pipeline by generating hypotheses for 3D object orientations from a scene crop. The Autoencoder architecture is trained to revert geometric and color input augmentations on random synthetic object views. This allows us to extract implicit representations of object orientations from real camera recordings.

[5] proposed to estimate orientation using regression directly on axis-angle and quaternion. Its loss function defines a geodesic distance between predicted and target rotation matrix or quaternion, and then simplifies it with Rodrigues' rotation formula. Because there are constraints in the representation and ambiguities in observed object poses, convergence issues appear [6], and thus regressing orientations in SO(3) is not practically usable in the generic case.

Extracting latent code in unsupervised manner seems to be a good alternative for a few reasons: It handles pose ambiguities directly and most importantly it has fast and robust feature extraction.

[7], [8] train an Autoencoder (AE) with input patches that only cover part of the object and therefore the latent code contain information of rotation and translation with respect to object center. A codebook is then used to store the latent-label code pair and provides nearest neighbors when queried. Pose is then estimated by a voting mechanism involved those nearest neighbors. This method is robust against occlusion because an input patch never cover the entire object and it ensembles patches with votes. However, it builds a huge codebook which is expensive to query.

We propose to train a similar AE-codebook combination but with complete object patches. The AE is fed with patches that cover the entire object and through various methods we predispose its latent code to encode the 3D rotation information. Moreover, we rely solely on synthetic training data and thus avoid the labeling problem of approaches like [9], [10].

[1] Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Germany. Email: martin.sundermeyer@dlr.de, [2] Department of Computer Science, Technical University of Munich (TUM), Germany. Email: rudolph.triebel@in.tum.de

[1] http://amzn.to/2v7HvAW

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

[11] showed that transferred and frozen feature extractor helps to mitigate over-fitting in object detection. To achieve better result, heavy data augmentations were applied to synthetic data. However, results differ across test images captured by different cameras suggests that there exist other type of over-fittings yet to be resolved.

[12] proposed an algorithm to train robust model against adversarial examples with variant of attacks such as iterative and least-likely Fast Gradient Sign Method (FGSM).

We will present several experiments exploring the feasibility of the above concepts.

III. Deep Orientation Estimation

In this chapter we derive a new training strategy for an encoder-decoder CNN-architecture to generate 3D object orientation hypotheses.

A. Autoencoder Variants

An AE is a variant of a Convolutional Neural Network (CNN) that is made of two parts: encoder and decoder. A typical encoder transforms, i.e. encodes, high dimensional input like an image into a low dimensional representation, i.e. a latent code. The decoder reconstructs the original input from this latent code.

There are several variants of AEs, aimed mainly to acquire various invariance properties and avoid over-fitting. Regularized AE proposed to impose weight decay which favors small weights to avoid over-fitting. Denoising AE [13] proposed to add random noise to the input image while maintaining a clean reconstruction target, so is to train an encoder that is robust to wider range of input. Both techniques implicitly encourage close-to-zero or sparse feature which is robust to input's variation.

In our approach, we use a strategy similar to denoising, but to enforce invariance against a wide range of appearance changes, save for changes in orientation, even going as far as being mostly invariant to real vs simulated data at the input. As detailed below, this will force the latent code to capture the orientation.

B. Translation Invariant Autoencoder

In order to specifically encode 3D object orientations, we have to control what the code represents and what input variations should be ignored. Therefore, we apply random augmentations to the input images against which the encoding shall become invariant. On the other hand, the reconstruction target remains to reconstruct a non-augmented ouput view. Since all augmentations are irrelevant for the reconstruction, the code does not contain any information about them. Using geometric augmentations, we can explicitly learn representations of object rotations independent of scale or translation. The input images are randomly translated while the output stays centered and such the encoding of the Augmented Autoencoder (AAE) becomes invariant to translation. This can also be applied to encode the whole SO(3) space of object views rendered from a 3D model (CAD or 3D reconstruction). It assures robustness against inaccurate object detections.

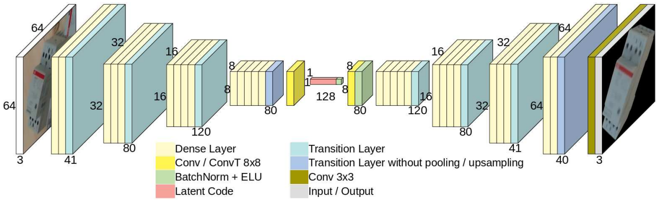

Fig. 2: Dense Autoencoder CNN architecture

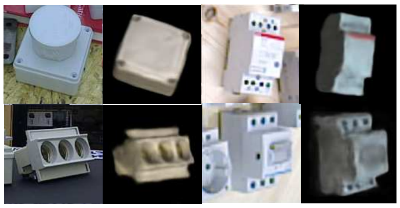

Fig. 3: AAE decoder reconstruction of T-LESS (left) and THR (right) crops

C. 3D Orientations from Synthetic Data

The AAE imposes a new Domain Randomization (DR) stategy to generate encodings invariant to irrelevant differences between real and simulated images. The goal is that the trained encoder treats the differences to real camera images as just another variation. We randomly apply color augmentations to the input training image, but leave the reconstruction target with black background and fixed light. The input object views are produced by rendering with random light positions and randomized diffuse and specular reflection [14]. We also use contrast, brightness, Gaussian blur and color distortions. To simulate occlusion we crop out squared parts of the object. Instead of black background we insert random background images from the Pascal VOC dataset [15].

D. Architecture and Training Procedure

Our baseline architecture is a convolutional Autoencoder with filter size 5x5. In our experiments we also use a densely connected AE that is depicted in Figure 2.

Using OpenGL, we render object views uniformly at random 3D orientations and constant distance along the camera axis (700mm). The resulting images are quadratically cropped and resized to $64 \times 64 \times 3$.

E. Codebook Generation and Lookup

Given the trained AAE, we are able to reconstruct a 3D object of a real scene crop for a variety of camera sensors (see Figure 3). In order to obtain the 3D orientation estimation of an object a so-called codebook has to be created.

Therefore we first of all generate clean – meaning centered and without augmentations on a black background – renderings from the object from equidistant viewpoints given a full view-sphere (based on a refined icosahedron [16]). In the next step, to cover the whole SO(3), each of the generated views are rotated in-plane at fixed intervals. Finally, all obtained views are forwarded to the AAE. The obtained latent code $z \in \mathcal{R}^{128}$ in combination with the corresponding rotation $R_{cam2obj} \in \mathcal{R}^{3x3}$ form the codebook, as can be seen in Figure 4 (blue encodings).

At test time, object crops from an RGB scene (e.g. coming from a 2D detector) are resized to match the input size and forwarded into the AAE. The obtained latent code (see

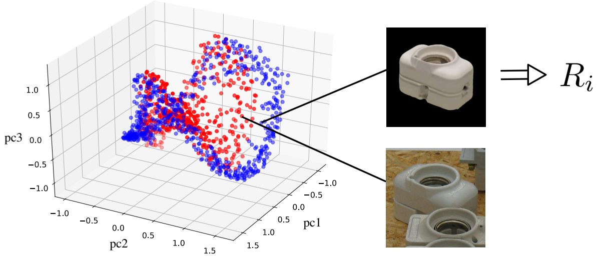

Fig. 4: Axes: 3 principle components pc_i of the latent space. Red: test encodings of scene 2, object 5, from the T-LESS dataset's [2] RGB images (captured by a Canon IXUS 950 IS). Blue: encodings of synthetic model views (upper hemisphere, upright)

Figure 4 red encodings), depicted as test code $z_{test} \in \mathcal{R}^{128}$, is then used to compute the cosine similarity against all codes $z_i \in \mathcal{R}^{128}$ from the codebook.

The highest similarities are determined in a k-Nearest-Neighbor (kNN) search and the corresponding rotation matrices $\{R_{kNN}\}$ from the codebook are returned as hypotheses of the 3D object orientation. The use of the cosine similarity is due to the computation efficiency on a single GPU for large codebooks (in our experiments 2562 viewpoints × 36 in-plane rotations result in 92232 total entries).

IV. Results and Comparisons

We evaluate the AAE on our THR dataset [17], parts of the T-LESS [2] dataset, and provide qualitative results on an IKEA cup. Since the 2D detection is not our focus here, we use the ground-truth crops of the objects from the datasets.

Our RGB-based pipeline is real-time capable, and will be combined with a compact bounding box detector for the demo using the Jetson TX2. On a modern desktop with GPU it runs above 40Hz[3] and the GPU memory easily holds multiple encoders and corresponding codebooks, thus allowing pose estimation of multiple objects in parallel. The live demo will feature the IKEA cup with fully synthetic training both for the detector and AAE.

A. Evaluation Criteria

1) Axis-angle Rotation Error: Our method is evaluated by computing the area under the e_R / recall curve (trapezoidal integration) AUC_{re} where $e_R \in [0^o, 180^o]$ defines the absolute angle error. This metric depicts an intuitive estimation of the object orientation error. It is convenient in scenarios where the exact orientation is uniquely defined. In case of objects with a symmetry axis (symmetries have to be predefined) we rectify the metric ($AUC_{re,rect}$) by considering both angle errors like: $min(|e_R|, |e_R - \pi|)$.

2) Visible Surface Discrepancy (e_{vsd}): This ambiguity-invariant pose error function [18] measures the pixel-wise depth deviation between the visible 3D surface of a rendered object model at ground truth pose and at the estimated pose. To individually evaluate the AAE we only predict the 3D orientation, here. Thereby, the issue of evaluating pose estimations resulting from symmetric object views is circumvented. The area under the e_{vsd} / recall curve (trapezoidal integration) AUC_{vsd} over multiple scene predictions reads

TABLE I: Ablation study on color augmentations for real vs synthetic data for training with testing on different test sensors. Object 5, all scenes, T-LESS [2], RGB only. Official SIXD metric (see subsection IV-A.2), with standard deviation of three runs in brackets.

Train	Test	dyn. light	add	contrast	multiply	invert	AUC_{vsd}
3D Reconstruction	Primesense	✓					0.472 (± 0.013)
		✓	✓				0.611 (± 0.030)
		✓	✓	✓			0.825 (± 0.015)
		✓	✓	✓	✓		0.876 (± 0.019)
		✓	✓	✓	✓	✓	**0.877** (± 0.005)
Primesense	Primesense		✓	✓	✓		0.890 (± 0.003)
3D Reconstruction	Kinect	✓					0.461 (± 0.022)
		✓	✓				0.580 (± 0.014)
		✓	✓	✓			0.701 (± 0.046)
		✓	✓	✓	✓		0.855 (± 0.016)
		✓	✓	✓	✓	✓	**0.897** (± 0.008)
Kinect	Kinect		✓	✓	✓		0.917 (± 0.007)

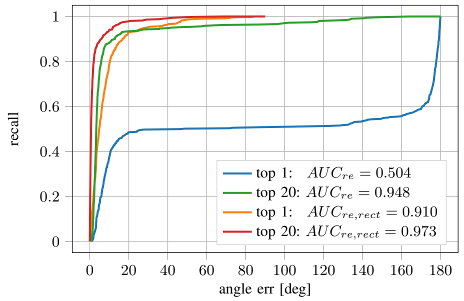

Fig. 5: Object 9 from T-LESS (has a symmetry axis) tested on all 504 Primesense RGB views of scene 11; recall at different cutoffs; with and without symmetry error correction; best of top 1 and top 20 predictions

B. Results

To assess the AAE alone, we predict the 3D orientation of object 5 from the T-LESS dataset on Primesense and Kinect RGB scene crops. Table I shows the influence of reverting different input augmentations. It can be seen that the effect of the color augmentations is cumulative. For texture-less objects, even the inversion of color channels seems to be beneficial since it prevents over-fitting to synthetic color information. Furthermore, training with real object recordings provided in T-LESS with random Pascal VOC background and augmentations yields only slightly better performance than the training with synthetic data.

Figure 5 shows typical e_R / recall curves for an axis symmetric object. While the nearest neighbor is likely to yield a good approximation of the 3D orientation, the top 20 nearest neighbors quite certainly include a correct estimate. This property of fast search space reduction is valuable for costly pose refinement methods. The latent space size is important for inference time. The performance increases with more dimensions and starts to saturate at $dim = 64$. Since inference is very efficient we chose $dim = 128$ in our experiments.

The Principal Component Analysis (PCA) in Figure 4 shows how synthetic and real object views are similarly encoded according to their orientation. Our DR strategy even allows the generalization from RGB views of an untextured CAD model. In that case, even more radical augmentations help but also slow down the convergence. Figure 6 shows the AAE predictions after being trained on views of an IKEA cup model.

C. Alternative Variants

We also adopted the Autoencoder using a DenseNet [19] architecture with growth $k = 40$ and 2/3/4/5 dense layers

[3] See preliminary demo preview running on a PC with an Nvidia GTX 1080: https://youtu.be/Z9pjmsJ-iiI

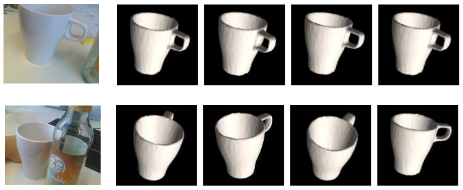

Fig. 6: Incomplete IKEA mug 3D orientation estimation from webcam stream (left), nearest training neighbors (right)

TABLE II: Effects of densely connected blocks, embedding loss (E) and adversarial augmentation (A) on top 1 $AUC_{re,rect}$ for TLESS scene 11. Baseline with ordinary 5x5 convolutional layers.

Object	Baseline	Dense	Dense^E	Dense^A	Dense^{EA}
5	0.896	0.928	0.916	0.934	0.935
8	0.923	0.915	0.910	0.930	0.913
9	0.909	0.890	0.883	0.904	0.905
10	0.848	0.860	0.851	0.882	0.887
Average	0.894	0.898	0.890	0.912	0.910

in 4 consecutive dense blocks for both encoder and decoder (mirrored). Dilated convolution [20] is used in every dense layer to increase the perception range. The amount of dilation grows from 1 to 3 and repeat in consecutive dense layers until the end of dense block. During training, cosine learning rate annealing with warm restarts [21] is used to achieve better parameters utilization. As a result, the size of DenseNet model is decreased to 3M parameters in the entire AE without negative effect in performance.

Furthermore, we tried an embedding loss that penalizes the deviation of latent codes between training data with and without augmentations using cosine similarity, so is to reduce variance in latent code induced by DR and therefore improve efficiency of decoder. On top of this we augment the synthetic input training data with adversarial examples generated using a randomized combination of different methods described in [12].

The results of the different variants are shown in Table II for a subset of the T-LESS data, and in Table III for the THR dataset. There is unfortunately no clear trend visible, as the performance varies on a case by case basis, while staying relatively similar. This suggests that the underlying principle of the AAE approach to enable the learning of a latent code that maps to SO(3) is not network dependent, and the amount and quality of the training data is the more crucial factor.

V. Conclusion

We introduced an augmented training method for a dense Autoencoder architecture that enables 3D object orientation estimation on RGB camera data. We showed that it is possible to solely train on rendered views of a 3D model. By reverting geometric and color input augmentations in the reconstruction, we learn representations that encode 3D object orientations independent of the domain, i.e. synthetic or real RGB images. Furthermore, pose ambiguities stemming from symmetric object views do not affect our approach since the self-supervised training is independent of pose labels.

TABLE III: $AUC_{re,rect}$ in the THR Dataset's test scenes (2 and 4) captured using an Intel SR300 (using only RGB).

Architecture:	Baseline AAE		Dense AAE	
Object	Scene 2	Scene 4	Scene 2	Scene 4
1: ABB_ESB24-40	0.789	0.770	0.792	0.785
2: EltakoNLZ12NP	0.833	0.748	0.818	0.823
3: EltakoS12-100	0.847	0.780	0.833	0.820
4: finder_dimmer	0.755	0.621	0.741	0.634
5: hagerCDA225D	0.756	0.609	0.743	0.565
6: hagerMBN116	0.694	0.642	0.636	0.645
7: hagerMBN316	0.809	0.743	0.812	0.795
8: hagerSH363N	0.765	0.753	0.771	0.771
9: Siemens5TE6800	0.792	0.639	0.672	0.581
Average	0.782	0.701	0.769	0.713

Acknowledgments

We thank Dr. Ingo Kossyk, Dimitri Henkel, and Manuel Brucker for helpful discussions.

References

[1] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1521–1529.

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

[3] J. Vidal, C.-Y. Lin, and R. Martí, "6d pose estimation using an improved method based on point pair features," *arXiv preprint arXiv:1802.08516*, 2018.

[4] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision*. Springer, 2016, pp. 21–37.

[5] S. Mahendran, H. Ali, and R. Vidal, "3d pose regression using convolutional neural networks," *arXiv preprint arXiv:1708.05628*, 2017.

[6] A. Saxena, J. Driemeyer, and A. Y. Ng, "Learning 3-d object orientation from images," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 794–800.

[7] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, "Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation," in *European Conference on Computer Vision*. Springer, 2016, pp. 205–220.

[8] V. A. Knyaz, O. Vygolov, V. V. Kniaz, Y. Vizilter, V. Gorbatsevich, T. Luhmann, N. Conen, W. Forstner, K. Khoshelham, S. Mahendran, *et al.*, "Deep learning of convolutional auto-encoder for image matching and 3d object reconstruction in the infrared range," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2155–2164.

[9] V. Balntas, A. Doumanoglou, C. Sahin, J. Sock, R. Kouskouridas, and T.-K. Kim, "Pose guided rgbd feature learning for 3d object pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3856–3864.

[10] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3d pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3109–3118.

[11] S. Hinterstoisser, V. Lepetit, P. Wohlhart, and K. Konolige, "On pre-trained image features and synthetic images for deep learning," *arXiv preprint arXiv:1710.10710*, 2017.

[12] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[13] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[14] B. T. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311–317, 1975.

[15] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[16] S. Hinterstoisser, S. Benhimane, V. Lepetit, P. Fua, and N. Navab, "Simultaneous recognition and homography extraction of local patches with a simple linear classifier," in *Proceedings of the British Machine Conference, pages*, 2008, pp. 10–1.

[17] M. Durner, S. Kriegel, S. Riedel, M. Brucker, Z.-C. Márton, F. Bálint-Benczédi, and R. Triebel, "Experience-based optimization of robotic perception," in *Advanced Robotics (ICAR), 2017 18th International Conference on*. IEEE, 2017, pp. 32–39.

[18] T. Hodaň, J. Matas, and Š. Obdržálek, "On evaluation of 6d object pose estimation," in *European Conference on Computer Vision*. Springer, 2016, pp. 606–619.

[19] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, vol. 1, no. 2, 2017, p. 3.

[20] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[21] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," 2016.