Supplement to "Pose Proposal Critic: Robust Pose Refinement by Learning Reprojection Errors"

Lucas Brynte
brynte@chalmers.se
Fredrik Kahl
fredrik.kahl@chalmers.se
Chalmers University of Technology
Gothenburg, SWEDEN

A Implementation Details

A.1 Pose Proposal Sampling

For training, we generate pose proposals by perturbing the ground-truth pose in three different ways: (1) With 30 % probability, a rotation around a random axis going through the object centre, whose magnitude is normally distributed with $\mu = 0$ and $\sigma = 45$ degrees. (2) With 30 % probability, a random lateral translation, normally distributed with $\mu = 0$ and $\sigma = 0.1d$, where d is the object diameter. (3) With 40 % probability, a relative depth perturbation, sampled from a log-normal distribution with $\mu = 0$ and $\sigma = \log 0.05$. This procedure and settings were found to work experimentally well.

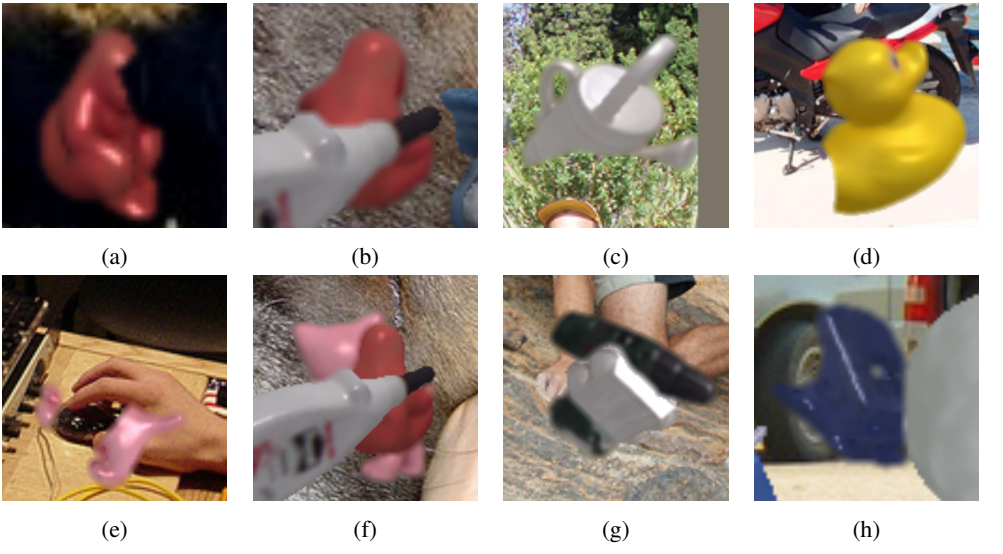
A.2 Rendering Synthetic Training Data

In addition to real annotated training images, we augment the training examples by rendering synthetic *observed* images, illustrated in Figure 1. Like for the pose proposals, rendering is done using OpenGL. Phong shading is applied and we noticed a performance boost by taking specular effects into account. In the spirit of Domain Randomization [10], we sample variations in light source position as well as shading parameters such as ambient / diffuse / specular weights, and the whiteness / shininess parameters of the specular effects. No perturbations are applied on albedo.

Random images from Pascal VOC2012 [2] were used as background and Gaussian blur was applied on the border in order to blend foreground and background and reduce overfitting to border artifacts as proposed by [1]. Gaussian blur was also applied to the whole object of interest as advised by [4].

Furthermore, occluding objects of other object categories are sometimes rendered in front of the object of interest. A visible region of at least 200 pixels is however ensured, otherwise occluders are resampled. In order to prevent overfitting towards the specific objects used for occlusion, occluded regions are replaced with background with a 50 % probability.

Finally, in the cases when we trained only on synthetic data, random noise in HSV-space was applied to the *observed* images.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 1: Synthetically rendered training examples of *observed* images. Occlusion is simulated by rendering additional objects in front of the object of interest, or alternatively the corresponding region is replaced with background, effectively making it transparent.

	ResNet-18 [3]	FlowNet 2.0 [5]
ADD(-S)-0.1D	50.92	**55.33**
REPROJ-S-5PX	62.66	**66.37**
5CM/5°-S	39.72	**41.52**

Table 1: Comparison of our method on Occlusion LINEMOD for different backbones.

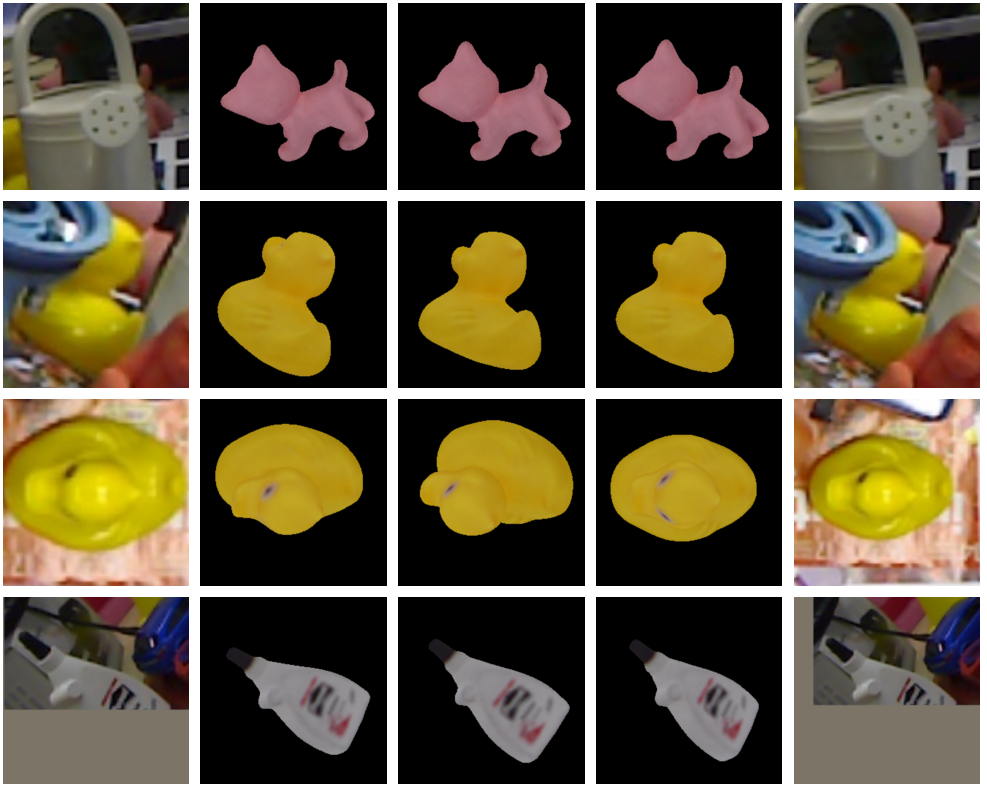
B Further Results

B.1 Backbone Comparison

As a first experiment, we evaluated the performance when altering the backbone on Occlusion LINEMOD. In addition to using the FlowNet 2.0 backbone [5], the encoder of Zakharov *et al*. [12], based on ResNet-18 [3] and a siamese network was re-implemented for comparison. As can be seen in Table 1, the FlowNet 2.0 model outperforms the alternative, giving further evidence for the conclusion made by Li *et al*. [6] that a feature extractor trained for optical flow is useful also for this task.

B.2 Illustration of Refinement Iterates

Figure 2 shows how our method gradually refines the pose for a few example frames of the Occlusion LINEMOD dataset, illustrated by the image patches of a few iterations. Despite the sub-optimal pose proposals from PVNet [9], the poses are accurately recovered.



(a) (b) (c) (d) (e)

Figure 2: Image patches during pose refinement iterations, for a few example frames of the Occlusion LINEMOD dataset. (b)-(d) show the *rendered* image patch for the initialization, the 10th iteration and the final iteration. (a) and (e) show the corresponding *observed* image patches for the initial as well as final pose. For an illustration of all iterations, we refer the reader to the supplied video, where in addition further examples are presented.

	Oberweger *et al.* [8]	PVNet [9]	PoseCNN [11] + DeepIM [6]	PVNet [9] + PPC (Ours)
ape	17.60	15.04	**59.18**	40.85
can	53.90	63.21	63.52	**82.44**
cat	3.31	20.30	26.24	**35.64**
driller	62.40	64.00	55.58	**71.33**
duck	19.20	33.86	**52.41**	49.08
eggbox*	25.90	43.32	**62.95**	57.28
glue*	39.60	49.83	**71.66**	62.90
holepuncher	21.30	41.40	**52.48**	43.14
Mean	30.40	41.37	**55.50**	55.33

Table 2: Results on Occlusion LINEMOD according to the ADD(-S)-0.1D metric.

	Oberweger *et al.* [8]	PVNet [9]	PoseCNN [11] + DeepIM [6]	PVNet [9] + PPC (Ours)
ape	**69.60**	66.84	69.02	68.97
can	82.60	**82.85**	56.14	79.29
cat	65.10	62.34	50.95	**66.47**
driller	73.80	70.68	52.94	**76.52**
duck	61.40	59.58	60.54	**66.93**
eggbox*	13.10	34.55	49.18	**49.28**
glue*	**54.90**	47.72	52.92	48.06
holepuncher	66.40	70.17	61.16	**75.45**
Mean	60.86	61.84	56.61	**66.37**

Table 3: Results on Occlusion LINEMOD according to the REPROJ-S-5PX metric. Note that [8] reports results according to REPROJ-5PX.

B.3 Detailed Pose Refinement Results

Here we present detailed (per-object) pose refinement results and corresponding comparison with other methods.

Tables 2, 3 and 4 show results on Occlusion LINEMOD for the ADD(-S)-0.1D, REPROJ-S-5PX and 5CM/5°-S metrics, respectively. The results of the corresponding experiments on synthetic data are reported in Tables 5, 6 and 7.

Similarly, results on LINEMOD are reported in Tables 8, 9 and 10, for the ADD(-S)-0.1D, REPROJ-5PX and 5CM/5° metrics, respectively.

The symmetric objects `eggbox` and `glue` are marked with *, and for them ADD(-S)-0.1D refers to ADD-S-0.1D, and the REPROJ-S-5PX and 5CM/5°-S metrics also take their ambiguities through 180 degree rotations around the "up"-axis into account.

	PVNet [9]	PoseCNN [11] + DeepIM [6]	PVNet [9] + PPC (Ours)
ape	37.18	**51.75**	47.69
can	63.38	35.82	**63.63**
cat	19.43	12.75	**33.19**
driller	60.21	45.24	**67.46**
duck	15.31	22.48	**23.01**
eggbox*	10.47	17.81	**33.87**
glue*	20.93	**42.73**	25.80
holepuncher	**40.00**	18.84	37.52
Mean	33.36	30.93	**41.52**

Table 4: Results on Occlusion LINEMOD according to the 5CM/5°-S metric. No results are reported by Oberweger *et al.* [8] on this metric.

	CDPN-synth [7]	CDPN-synth [7] + PPC-synth (Ours)
ape	17.65	**29.95**
can	13.57	**36.68**
cat	14.29	**16.84**
driller	5.00	**12.50**
duck	**20.74**	20.21
eggbox*	**33.16**	**33.16**
glue*	26.62	**29.87**
holepuncher	**24.00**	9.50
Mean	18.76	**23.59**

Table 5: Synthetic results on Occlusion LINEMOD according to the ADD(-S)-0.1D metric.

	CDPN-synth [7]	CDPN-synth [7] + PPC-synth (Ours)
ape	48.66	**59.89**
can	24.62	**34.17**
cat	35.20	**40.82**
driller	7.50	**15.00**
duck	51.60	**54.79**
eggbox*	34.20	**34.72**
glue*	**14.94**	12.99
holepuncher	**48.50**	35.50
Mean	32.22	**35.99**

Table 6: Synthetic results on Occlusion LINEMOD according to the REPROJ-S-5PX metric.

	CDPN-synth [7]	CDPN-synth [7] + PPC-synth (Ours)
ape	26.74	**36.90**
can	17.59	**26.13**
cat	13.78	**18.88**
driller	6.50	**11.00**
duck	15.43	**16.49**
eggbox*	**30.57**	23.83
glue*	5.84	**9.74**
holepuncher	**19.00**	15.50
Mean	16.13	**19.81**

Table 7: Synthetic results on Occlusion LINEMOD according to the 5CM/5°-S metric.

	PoseCNN [11]	PoseCNN [11] + DeepIM [6]	PoseCNN [11] + PPC (Ours)
ape	27.71	**76.95**	75.14
benchvise	68.87	**97.48**	94.28
camera	47.35	93.53	**96.18**
can	71.33	92.81	**96.95**
cat	56.64	82.14	**89.82**
driller	65.28	94.95	**97.92**
duck	42.86	**77.65**	69.39
eggbox*	97.84	97.09	**98.59**
glue*	94.88	**99.42**	92.95
holepuncher	44.00	52.81	**68.70**
iron	65.47	**98.26**	90.19
lamp	69.96	97.50	**98.27**
phone	54.39	**87.72**	84.32
Mean	62.04	88.33	**88.67**

Table 8: Results on LINEMOD according to the ADD(-S)-0.1D metric.

	PoseCNN [11]	PoseCNN [11] + DeepIM [6]	PoseCNN [11] + PPC (Ours)
ape	82.67	**98.38**	97.71
benchvise	49.95	96.99	**97.87**
camera	71.67	**98.92**	98.63
can	69.85	**99.70**	97.64
cat	92.01	98.70	**98.80**
driller	43.45	96.13	**97.13**
duck	91.73	**98.5**	97.93
eggbox*	41.82	96.15	**98.50**
glue*	87.73	**98.94**	96.24
holepuncher	59.52	96.29	**98.57**
iron	41.68	**97.24**	**97.24**
lamp	48.27	**94.24**	94.15
phone	58.46	97.73	**98.39**
Mean	64.52	97.53	**97.60**

Table 9: Results on LINEMOD according to the REPROJ-5PX metric. Note that [6] reports results according to REPROJ-S-5PX.

	PoseCNN [11]	PoseCNN [11] + DeepIM [6]	PoseCNN [11] + PPC (Ours)
ape	6.95	90.38	**96.48**
benchvise	13.58	88.65	**90.40**
camera	20.39	**95.78**	91.67
can	24.39	92.81	**94.39**
cat	24.98	87.62	**96.01**
driller	18.25	92.86	**96.13**
duck	18.23	**85.16**	83.10
eggbox*	16.53	63.85	**95.49**
glue*	19.50	**83.01**	73.07
holepuncher	15.81	54.52	**82.11**
iron	12.97	**92.65**	92.03
lamp	24.38	90.88	**92.51**
phone	19.26	**89.16**	83.29
Mean	18.14	85.21	**89.74**

Table 10: Results on LINEMOD according to the 5CM/5° metric. Note that [6] reports results according to 5CM/5°-S.

C Additional Notes

C.1 Negative Depth Correction of Pose Proposals

We observed that the pose proposals from PVNet [9] sometimes have negative depth, and in this case we switched sign for the object center position (in the camera frame), and rotated the object 180 degrees around the principal axis of the camera, in order to yield a feasible estimate with similar projection (the projection is identical for points on the plane which goes through the object center and is parallel to the principal plane of the camera). This correction is done both when reporting the results of [9], and when reporting the results of our refinement.

References

[1] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[2] 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.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[4] Stefan Hinterstoisser, Vincent Lepetit, Paul Wohlhart, and Kurt Konolige. On pre-trained image features and synthetic images for deep learning. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[5] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[6] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep iterative matching for 6d pose estimation. *International Journal of Computer Vision*, 128 (3):657–678, November 2019. doi: 10.1007/s11263-019-01250-9. URL https://doi.org/10.1007/s11263-019-01250-9.

[7] Zhigang Li, Gu Wang, and Xiangyang Ji. CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[8] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3D object pose estimation. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[9] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise voting network for 6DoF pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[10] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017.

[11] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. *Robotics: Science and Systems (RSS)*, 2018.

[12] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. DPOD: 6D pose object detector and refiner. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.