Supplementary Material of Self6D: Self-Supervised Monocular 6D Object Pose Estimation

Gu Wang[1,2,*] Fabian Manhardt[2,*] Jianzhun Shao[1]
Xiangyang Ji[1] Nassir Navab[2] Federico Tombari[2,3]

[1]Tsinghua University, BNRist [2]Technical University of Munich [3]Google
{wangg16, sjz18}@mails.tsinghua.edu.cn, xyji@tsinghua.edu.cn,
{fabian.manhardt, nassir.navab}@tum.de, tombari@in.tum.de

Abstract. This document supplements our main paper entitled *Self6D: Self-Supervised Monocular 6D Object Pose Estimation* by providing i) more details on the architecture, employed hyper-parameters and experimental setup, ii) detailed analysis on the quality of predicted masks, iii) illustrating additional qualitative results for all conducted experiments and iv) results for a subset of *YCB-Video*. Finally, we v) provide a video, depicting our method and showing additional results.

1 Implementation Details

1.1 More Architecture Details

As mentioned in Section 3 of the main paper, we employ different lightweight branches to predict the following outputs from the fused FPN feature maps: the 3D rotation R parameterized as a 4D quaternion q, the 3D translation t constituted by the 2D projection (c_x, c_y) of the 3D object centroid together with the distance z, and the visible object mask M^P.

For fusion of the FPN feature maps, we first apply a convolutional layer to reduce the dimension of each layer from 128 to 64, we then conduct bilinearly rescaling to each layer to make them spatially equal with the largest feature map (*i.e.*, 60×80). Finally, we concatenate all different levels to obtain the fused FPN feature map.

Each of our lightweight branches consists of 4 convolutional layers (except 2 for the 2D centroid) with Group Normalization [13] and Leaky ReLU [15] having a negative slope of 0.1. The final output of each predictor is obtained by employing either a 3×3 convolutional layer (*mask*) or a fully connected layer employed after flattening the output of forelast layer (*centroid*, *distance* and *rotation*).

In line with other works [16,7], we freeze the parameters in the first 3 stages of the backbone, to reduce the risk of overfitting to synthetic data. We additionally apply various augmentations to the training RGB images, similar to [9,5].

* Equal contribution.

hyper-parameter	λ_{class}	λ_{box}	λ_{mask}	λ_{pose}	α	β	γ	η
value	1.0	1.0	1.0	10.0	0.2	1.0	0.15	100

Table 1: Employed hyper-parameters for losses.

1.2 Experimental Setup

We implemented our method in PyTorch 1.3 [8] and ran all our experiments on a NVIDIA TitanX GPU. In the first stage, we trained our model with a batch size of 12 for 8 epochs. During self-supervision, we trained the model for another 100 epochs with a batch size of 3. We utilized RAdam [6] combined with Lookahead [17] for optimization. The initial learning rate was set to 10^{-4} and decayed after 72% of the training phase using a cosine schedule [3].

Employed Hyper-Parameters for Losses. Table 1 shows the employed hyper-parameters for training with synthetic data

$$\mathcal{L}_{synthetic} \coloneqq \lambda_{class}\mathcal{L}_{focal} + \lambda_{box}\mathcal{L}_{giou} + \lambda_{mask}\mathcal{L}_{bce} + \lambda_{pose}\mathcal{L}_{pose}, \tag{1}$$

and self-supervised training on real data

$$\mathcal{L}_{Self} = \mathcal{L}_{visual} + \eta\mathcal{L}_{geom}, \tag{2}$$

with

$$\mathcal{L}_{visual} = \mathcal{L}_{mask} + \alpha\mathcal{L}_{ab} + \beta\mathcal{L}_{ms\text{-}ssim} + \gamma\mathcal{L}_{perceptual}. \tag{3}$$

The hyper-parameters are empirically chosen to level the different loss contributions.

1.3 Details of $\mathcal{L}_{ms\text{-}ssim}$

In this section, we present the details of MS-SSIM loss ($\mathcal{L}_{ms\text{-}ssim}$), which is an important part for the visual alignment ($\mathcal{L}_{visual}$) with respect to our proposed self-supervision ($\mathcal{L}_{Self}$). The structural similarity index (SSIM) [11] for pixel p is defined as

$$ssim(p) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \cdot \frac{2\sigma_{xy} + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} = l(p) \cdot cs(p). \tag{4}$$

Thereby, SSIM is computed on blocks, with (μ_x, μ_y) and $\left(\begin{smallmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{smallmatrix}\right)$ denoting the corresponding means and covariance matrix with respect to p, respectively. In practice, we employ two constants $c_1 = 0.01$ and $c_2 = 0.03$ for numerical stability. In [18], the authors extend this measurement to a perceptually-motivated loss

	Ape	Bvise	Cam	Can	Cat	Drill	Duck	Eggbox	Glue	Holep	Iron	Lamp	Phone	Mean
F1 score	94.44	92.04	81.57	92.51	91.60	91.51	92.17	94.82	89.57	91.02	91.31	78.16	84.52	89.63
IoU	94.72	92.45	83.41	92.88	92.08	92.03	93.76	95.02	90.35	91.90	91.80	79.13	85.39	90.38

Table 2: **Detailed evaluation of predicted masks.** We report the F1 score(%) and IoU(%) on *LineMOD* test set.

function referring to a widely used multi-scale version of SSIM, namely MS-SSIM. Given a dyadic pyramid of s levels, MS-SSIM can be written as

$$ms\text{-}ssim(p) = l_s^{\alpha}(p) \cdot \prod_{j=1}^{s} cs_j^{\beta_j}(p), \tag{5}$$

where l_s and cs_j are the functions defined on the right in Eq. (4) at scale s and j, respectively. As in [18], we set $\alpha = 1$, $\beta_j = 1$, $\forall j \in \{1, ..., s\}$, and $s = 5$. For convenience, given two RGB images I_A and I_B, we designate the MS-SSIM between them as $ms\text{-}ssim(I_A, I_B, s)$. Since, $ms\text{-}ssim$ measure the similarity between two samples with 1 being the maximum, we rewrite the loss with respect to MS-SSIM as

$$\mathcal{L}_{ms\text{-}ssim} \coloneqq 1 - ms\text{-}ssim(I_A, I_B, s). \tag{6}$$

2 Detailed Analysis on the Quality of Predicted Masks

Since our self-supervision relies on high quality of the predicted masks from our synthetically trained model Self6D-LB, we present the detailed quantitative evaluation in Table 2. Specifically, we calculated F1 score and IoU between the predicted masks and the ground-truth masks on *LineMOD* test set. We can see that the predicted masks are very accurate on almost all objects, with the average F1 score 89.63% and mIoU 90.38%. Thus we can regard the predicted masks from Self6D-LB as a reliable self-supervision signal. Fig. 1 shows some qualitative examples for the predicted masks on *LineMOD* test set.

Fig. 1: Qualitative results for predicted masks on *LineMOD* test set. (Best viewed zoomed-in, in color).

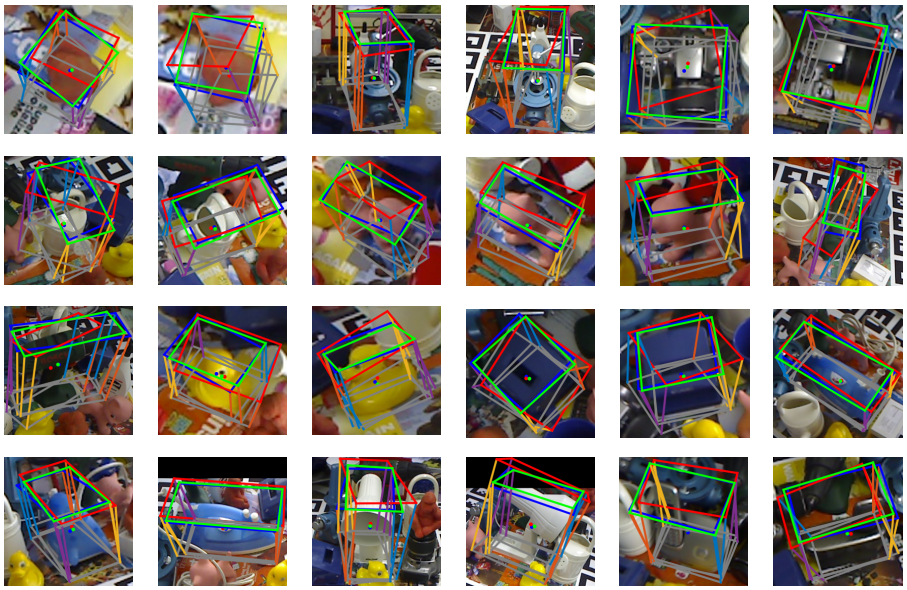

Fig. 2: Qualitative results on *Cropped LineMOD*. We visualize the 6D pose by overlaying the image with the corresponding transformed 3D bounding box. While *Blue* constitutes the ground truth pose, we demonstrate in *Red* and *Green* the results before and after applying our self-supervision, respectively.

3 More Qualitative Results

In this section, we want to present additional qualitative results for each conducted experiment.

Domain Transfer. In Fig. 2, we demonstrate our results for domain transfer, leveraging the *Cropped LineMOD* dataset [12]. While we constitute the ground truth poses in *Blue*, we demonstrate in *Red* and *Green* the results before (Self6D-LB) and after applying our self-supervision (Self6D), respectively.

6D Pose Estimation. We additionally present qualitative results for 6D pose estimation. Notice that we again depict the ground truth pose in *Blue* and the prediction before (Self6D-LB) and after self-supervision (Self6D) in *Red* and *Green*. While the initial results are very noisy in terms of 6D pose, the self-supervised model produces highly accurate pose estimates with respect to the given ground truth.

The results on *LineMOD Occlusion* [1], *HomebrewedDB* [4], and *LineMOD* [2] are respectively depicted in Fig. 3, Fig. 4, and Fig. 5.

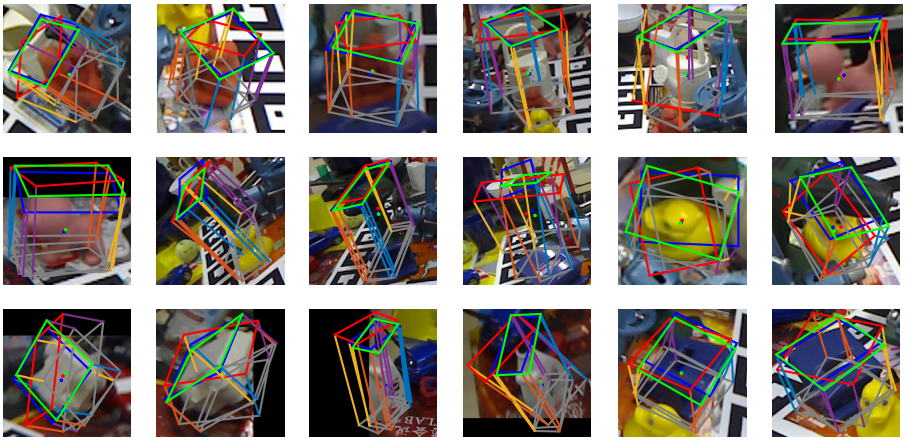

Fig. 3: Qualitative results on *LineMOD Occlusion*. We visualize the 6D pose by overlaying the image with the corresponding transformed 3D bounding box. While *Blue* constitutes the ground truth pose, we demonstrate in *Red* and *Green* the results before and after applying our self-supervision, respectively.

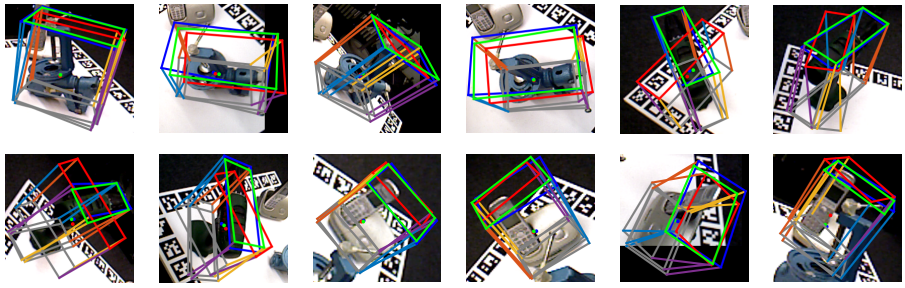

Fig. 4: Qualitative results on *HomebrewedDB*. We visualize the 6D pose by overlaying the image with the corresponding transformed 3D bounding box. While *Blue* constitutes the ground truth pose, we demonstrate in *Red* and *Green* the results before and after applying our self-supervision, respectively.

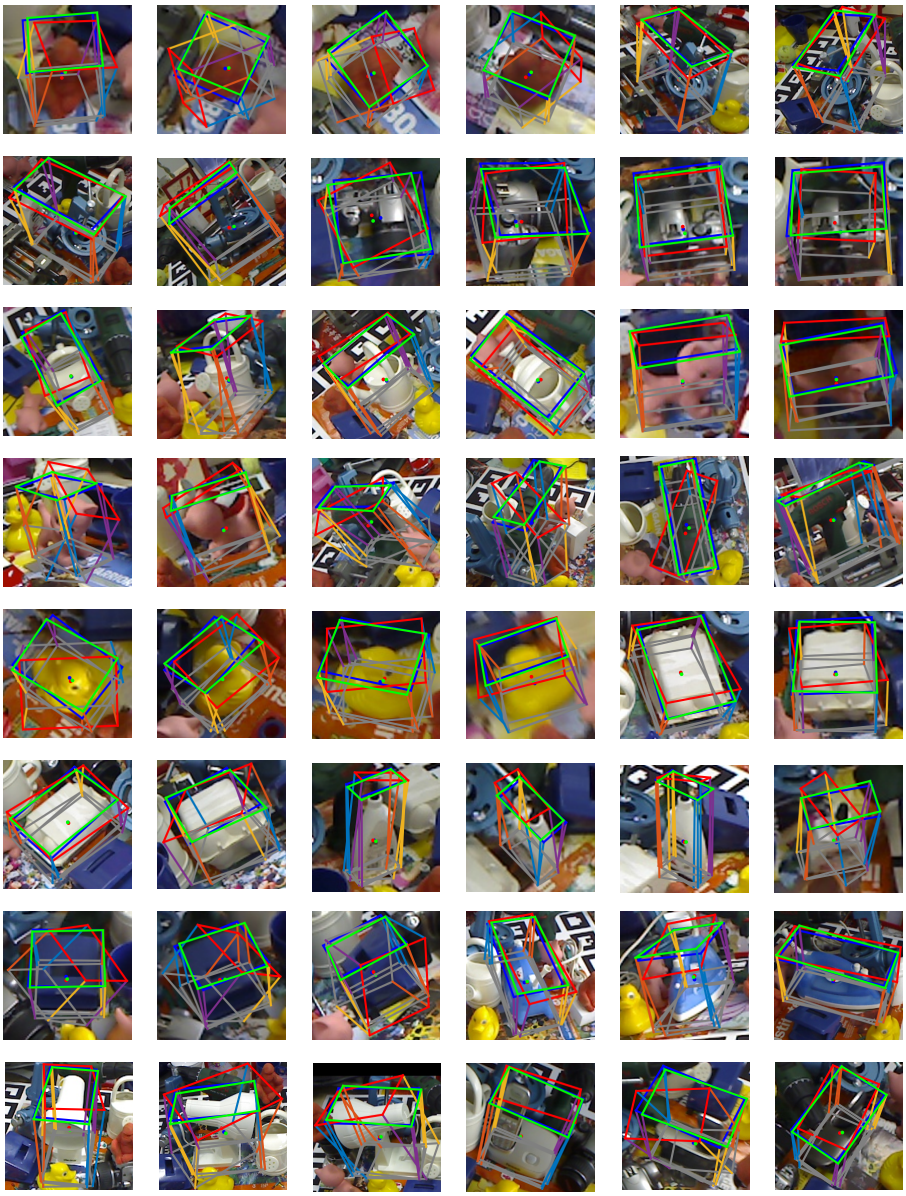

Fig. 5: More qualitative results on *LineMOD*. We visualize the 6D pose by overlaying the image with the corresponding transformed 3D bounding box. While *Blue* constitutes the ground truth pose, we demonstrate in *Red* and *Green* the results before and after applying our self-supervision, respectively.

Self-Supervision *v.s.* 6D Pose Error. In the main paper, we demonstrate the existence of a strong correlation between our self-supervised loss function and the 6D pose accuracy. In particular, we conduct our self-supervision separately to single images for in total 200 iterations. In Fig. 6, we illustrate additional qualitative results for this experiment. Despite not possessing any pose labels, we can always compute pose estimates (*Green*) which are almost perfectly aligned with the ground truth (*Blue*). This is even true for poses, which are badly initialized using Self6D-LB (*Red*). This strongly suggest that our loss function is able to solve for the 6D pose, while actually optimizing for visual and geometric alignment.

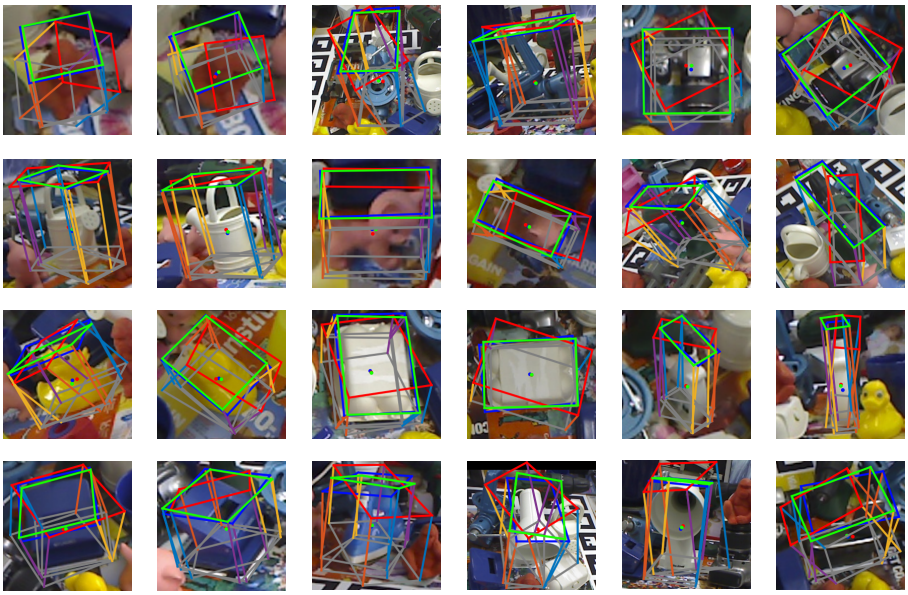

Fig. 6: More qualitative results on *Self-Supervision v.s. 6D Pose Error*. We visualize the 6D pose by overlaying the image with the corresponding transformed 3D bounding box. While *Blue* constitutes the ground truth pose, we demonstrate in *Red* and *Green* the results before and after applying our self-supervision to the single image for 200 iterations, respectively.

4 Results on *YCB-Video*

	Self6D-LB	Self6D
006_mustard_bottle	73.7	**88.2**
007_tuna_fish_can	26.6	**69.7**
011_banana	4.0	**10.3**
025_mug	23.9	**43.4**
035_power_drill	21.4	**31.4**
Mean	29.9	**48.6**

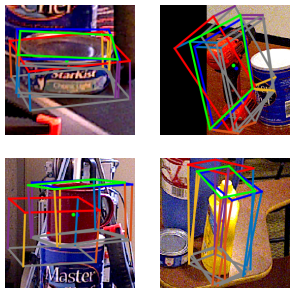

Table 3: *Left*: Results on *YCB-Video* [14] dataset. We report Average Recall (%) of ADD-S for the top 4 objects and ADD for 035_power_drill. *Right*: Qualitative results.

Although being out of the scope for the main body of this work, we further evaluated our method on a subset (5 objects) of *YCB-Video* [14], in order to emphasize that Self6D can be applied to any 6D pose scenario.

We first fully supervise Self6D-LB on a mixture of synthetic RGB images from *YCB-Video* and photorealistic renderings from [10]. We then self-supervise the model on 10% of the real RGB-D images for each of these 5 objects from the training set, however, without leveraging any 6D annotations (Self6D). Since the first four objects exhibit rotational symmetries, we report the average recall for ADD-S for them. For the remaining object (*i.e. 035_power_drill*), we refer to ADD instead, as it does not possess rotational symmetries. In line with all other experiments, we can constitute that leveraging our self-supervision, we are able to significantly enhance the average recall with respect to ADD(-S) for each object (Table 3).

References

1. Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6D object pose estimation using 3D object coordinates. In: ECCV. pp. 536–551 (2014)
2. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: ACCV. pp. 548–562 (2012)
3. Ilya Loshchilov, F.H.: SGDR: stochastic gradient descent with warm restarts. In: ICLR (2017)
4. Kaskman, R., Zakharov, S., Shugurov, I., Ilic, S.: HomebrewedDB: RGB-D dataset for 6d pose estimation of 3d objects. In: ICCVW (2019)
5. Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In: ICCV. pp. 1521–1529 (2017)
6. Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Han, J.: On the variance of the adaptive learning rate and beyond. In: ICLR (April 2020)

7. Manhardt, F., Kehl, W., Gaidon, A.: Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In: CVPR (2019)
8. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS. pp. 8026–8037 (2019)
9. Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R.: Implicit 3d orientation learning for 6d object detection from rgb images. In: ECCV. pp. 699–715 (2018)
10. Tremblay, J., To, T., Birchfield, S.: Falling things: A synthetic dataset for 3d object detection and pose estimation. In: CVPRW. pp. 2038–2041 (2018)
11. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing (TIP) **13**(4), 600–612 (2004)
12. Wohlhart, P., Lepetit, V.: Learning descriptors for object recognition and 3d pose estimation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3109–3118 (2015)
13. Wu, Y., He, K.: Group normalization. In: ECCV. pp. 3–19 (2018)
14. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. RSS (2018)
15. Xu, B., Wang, N., Chen, T., Li, M.: Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853 (2015)
16. Zakharov, S., Shugurov, I., Ilic, S.: Dpod: 6d pose object detector and refiner. In: ICCV. pp. 1941–1950 (2019)
17. Zhang, M., Lucas, J., Ba, J., Hinton, G.E.: Lookahead optimizer: k steps forward, 1 step back. In: NeurIPS. pp. 9593–9604 (2019)
18. Zhao, H., Gallo, O., Frosio, I., Kautz, J.: Loss functions for image restoration with neural networks. IEEE Transactions on Computational Imaging **3**(1), 47–57 (2016)