MRC-Net: 6-DoF Pose Estimation with MultiScale Residual Correlation Supplementary Material

Yuelong Li *
Amazon Inc.
yuell@amazon.com

Yafei Mao *
Amazon Inc.
yafeimao@amazon.com

Raja Bala
Amazon Inc.
rajabl@amazon.com

Sunil Hadap
Amazon Inc.
hadsunil@lab126.com

1. Implementation Details

1.1. Network Architecture

Referring to Fig. 2 in the main paper, we input a 256×256 crop into a ResNet-34 backbone without pooling and fully-connected layers, yielding a $16 \times 16 \times 512$ feature tensor. The decoder conducts two rounds of $2\times$ upsampling on this tensor, incorporating low-level features via skip connections, resulting in a 64×64 feature tensor. Following the approach of [2, 6, 18], we use individual decoders for each object, each producing a 64-channel image feature and a binary segmentation mask indicating object visibility. These outputs are concatenated and downsampled through two consecutive convolution layers with a stride of two, forming a three-scale feature pyramid. This is processed through an ASPP block [3] with dilation rates $r = 2, 3, 4, 6, 8, 12$, aligning with the rates in the ASPP block preceding the regression and classification heads, and capturing both local and global context. We do not track running means and variances across shared batch normalization layers, since real and rendered batches can have different statistics.

Input resolution	Input channels	Output channels	Stride
64×64	186	128	2
32×32	377	256	2
16×16	505	256	1

Table 1. Architecture details of the 3×3 convolutions inside the MRC block. Each convolution layer is followed by group normalization [21] and the Swish activation function [15].

The feature pyramids from both real and rendered images serve as inputs to our novel MRC block. All 1×1 convolutions within this block have both 128 input and output channels. Additionally, a sequence of 3×3 convolutions is employed to fuse real and correlation features. The architecture of these convolution layers is detailed in Table 1.

Finally, each individual task head comprises a simple two-layer multi-layer perceptron (MLP) with a hidden size of 256.

1.2. Rotation and Translation Representations

To form a uniform partition of $SO(3)$, we follow the approach in [23] to generate $K = 4608$ uniform grids in $SO(3)$. This involves using Hopf coordinates [23] to decompose 3D rotation into a 1D in-plane rotation (around the z axis) and a 2D out-of-plane rotation. We then generate $m_2 = 192$ out-of-plane components using the formula from [5] with $N_{\text{side}} = 4$, followed by $m_1 = \sqrt{\pi m_2} \approx 24$ in-plane components.

For translation, we adopt the SITE format [13] to disambiguate the network prediction target based on local image patches. Our formulation of τ_z slightly differs from the original version:

$$\tau_z = \frac{t_z}{r\sqrt{f_x f_y}},$$

where f_x, f_y are the camera focal lengths and r is the resizing ratio of the bounding box. This adjustment is made to normalize τ_z consistently across different cameras.

To measure distance between poses, we use the formulation in [10] which computes vertex-based distances over CAD models. To ensure differentiability near the origin, we adopt smooth L1 loss [4] as a substitute of vertex distances. We also leverage the symmetry-aware formulation in [10] to account for object symmetries.

To decouple relative rotation and translation between real and rendered images,we closely follow the approach of [10, 12], differing only in the use of $\Delta\tau_z$ as an *additive* residual for τ_z instead of a *multiplicative* one. This modification has been found to enhance training stability.

*Equal contribution

2. Additional Ablation Studies

MRC-Net uses a Siamese structure for classification and regression branches. This enables real and rendered image features to be projected into a shared embedding space to accurately capture correspondences between them. To verify efficacy of this design choice, we conduct ablation experiments comparing Siamese and non-Siamese versions. The latter decouples the network weights in the classification and regression branches, as done in [11]. We use similar strategies as [11, 24] to train classifier and regressor separately. During training, the ground truth pose is perturbed synthetically by adding random noise to simulate classification errors. We then render the images based on these noisy poses. We follow the same noise schedule as described in [11]. Comparisons across different methods are summarized in Table 2a.

The non-Siamese network with separate training exhibits a significant 11.4% drop in average recall (AR) compared to the proposed Siamese model. Even with an end-to-end training strategy, it lags notably by 1.2%. In contrast, the Siamese model achieves not only superior performance but also computational and storage efficiency by nearly halving parameters through parameter sharing.

We next study the impact of the number of rotation buckets K. Our choice $K = 4608 (N_{\text{side}} = 4)$ is the maximum value that we could fit into our GPU memory, which gives a relative angle of $14.7°$ between adjacent rotation buckets [5]. In our $SO(3)$ partitioning scheme (Section 1.2), alternative values for K include $K = 576 (N_{\text{side}} = 2)$ and $K = 1944 (N_{\text{side}} = 3)$. Results are summarized in Table 2b. As K increases, performance of our model improves slightly, with $K = 4608$ achieving the highest AR scores. This validates our choice of K and supports the assumption that finer classification can facilitate regression by providing a better pose initialization.

3. Detailed Results on YCB-V

Table 3 shows a comprehensive breakdown of per-object results for the AUC of ADD-S and AUC of ADD(-S). Previous works commonly use Mask RCNN detections provided by CosyPose [10] or FCOS detections provided by CDPN [13] for their evaluations. Therefore, we present results for both. Compared to previous techniques, our method demonstrates notable improvements across most objects, resulting in an increase of +5.1% in the average ADD-S AUC and +7.9% in the average ADD(-S) AUC when combined with FCOS detections.

Further insights into the ADD(-S) metric for individual objects are presented in Table 4. MRC-Net outperforms prior models across various objects, leading to an +1.8% improvement in average ADD(-S) recall when using FCOS detections.

4. Additional Qualitative Evaluation

Visual examples from the T-LESS, YCB-V and LM-O datasets are presented in Fig. 1, Fig. 2 and Fig. 3, respectively. These examples demonstrate MRC-Net's ability to accurately predicting object poses in challenging scenarios such as heavy occlusions, diverse viewpoints, and distracting background clutter.

We present typical failure cases per dataset in Fig. 4. In Fig. 4 (b), the object in the center has an inaccurately estimated pose due to heavy occlusion. In Fig. 4 (d), the object 0024_bowl in the training set predominantly faces upward, and occasional incorrect flips of the bowl occur due to such pose imbalance. Fig. 4 (f) illustrates the eggbox object with incorrect rotations, partly attributed to inaccuracies in the CAD model.

Method	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	AR
Non-Siamese network with separate training	56.1	63.5	77.4	65.7
Non-Siamese network with end-to-end training	68.9	73.0	85.8	75.9
Siamese network (Ours)	70.6	74.7	86.0	77.1

(a) Ablation studies on Siamese and non-Siamese designs.

K	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	AR
576	70.3	74.3	85.6	76.7
1944	70.4	74.4	85.8	76.9
4608	70.6	74.7	86.0	77.1

(b) Ablation studies on K, the number of rotation buckets.

Table 2. Additional ablation studies. Models are trained on T-LESS dataset using synthetic training set [8].

Method	PoseCNN [22]		GDR-Net [20]		ZebraPose [17]		CheckerPose [14]		MRC-Net		MRC-Net	
Detection	Built-in		Faster-RCNN [13, 16]		FCOS [13, 19]		FCOS [13, 19]		Mask RCNN [7, 10]		FCOS [13, 19]	
Metric	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)
002_master_chef_can	84.0	50.9	96.3	65.2	93.7	75.4	87.5	67.7	98.2	98.2	98.0	98.0
003_cracker_box	76.9	51.7	97.0	88.8	93.0	87.8	93.2	86.7	100.0	98.6	99.9	98.8
004_sugar_box	84.3	68.6	98.9	95.0	95.1	90.9	95.9	91.7	100.0	98.9	100.0	99.0
005_tomato_soup_can	80.9	66.0	96.5	91.9	94.4	90.1	94.0	89.9	95.1	92.6	96.0	93.5
006_mustard_bottle	90.2	79.9	100.0	92.8	96.0	92.6	95.7	90.9	99.9	98.9	99.7	98.1
007_tuna_fish_can	87.9	70.4	99.4	94.2	96.9	92.6	97.5	94.4	97.2	87.3	97.3	87.5
008_pudding_box	79.0	62.9	64.6	44.7	97.2	95.3	94.9	91.5	99.6	98.4	99.2	97.9
009_gelatin_box	87.1	75.2	97.1	92.5	96.8	94.8	96.1	93.4	98.9	96.7	98.9	96.3
010_potted_meat_can	78.5	59.6	86.0	80.2	91.7	83.6	86.4	80.4	79.2	75.2	84.1	79.0
011_banana	85.9	72.3	96.3	85.8	92.6	84.6	95.7	90.1	98.9	93.0	98.3	91.9
019_pitcher_base	76.8	52.5	99.9	98.5	96.4	93.4	95.8	91.9	99.9	99.6	100.0	99.3
021_bleach_cleanser	71.9	50.5	94.2	84.3	89.5	80.0	90.6	83.2	92.5	83.2	93.7	85.2
024_bowl	69.7	69.7	85.7	85.7	37.1	37.1	82.5	82.5	99.0	99.0	98.7	98.7
025_mug	78.0	57.7	99.6	94.0	96.1	90.8	96.9	92.7	99.7	99.7	99.4	99.4
035_power_drill	72.8	55.1	97.5	90.1	95.0	89.7	94.7	88.8	99.9	98.1	99.8	97.9
036_wood_block	65.8	65.8	82.5	82.5	84.5	84.5	68.3	68.3	83.9	83.9	84.4	84.4
037_scissors	56.2	35.8	63.8	49.5	92.5	84.5	91.7	81.6	90.4	78.0	94.5	85.9
040_large_marker	71.4	58.0	88.0	76.1	80.4	69.5	83.3	72.3	96.4	96.4	97.3	97.3
051_large_clamp	49.9	49.9	89.3	89.3	85.6	85.6	90.0	90.0	93.2	93.2	97.1	97.1
052_extra_large_clamp	47.0	47.0	93.5	93.5	92.5	92.5	91.6	91.6	62.1	62.1	98.3	98.3
061_foam_brick	87.8	87.8	96.9	96.9	95.3	95.3	94.1	94.1	95.5	95.5	96.7	96.7
Mean	75.9	61.3	91.6	84.3	90.1	85.3	91.3	86.4	94.3	91.7	96.7	94.3

Table 3. Detailed results on YCB-V [22] w.r.t. AUC of ADD-S and AUC of ADD(-S). We highlight the best AUC of ADD-S results in red, and the best AUC of ADD(-S) in blue.

Method	SegDriven [9]	GDR [20]	Zebra [17]	CheckerPose [14]	MRC-Net	MRC-Net
Detection	Built-in	Faster RCNN [13, 16]	FCOS [13, 19]	FCOS [13, 19]	Mask RCNN [7, 10]	FCOS [13, 19]
002_master_chef_can	33.0	41.5	62.6	45.9	**96.3**	94.3
003_cracker_box	44.6	83.2	98.5	94.2	**100.0**	**100.0**
004_sugar_box	75.6	91.5	96.3	98.3	**100.0**	**100.0**
005_tomato_soup_can	40.8	65.9	80.5	83.2	79.9	**81.0**
006_mustard_bottle	70.6	90.2	**100.0**	99.2	98.0	96.7
007_tuna_fish_can	18.1	44.2	70.5	**88.9**	14.3	17.0
008_pudding_box	12.2	2.8	**99.5**	86.5	92.0	93.3
009_gelatin_box	59.4	61.7	**97.2**	86.0	69.3	70.7
010_potted_meat_can	33.3	64.9	**76.9**	70.0	61.3	62.2
011_banana	16.6	64.1	71.2	**96.0**	86.0	82.7
019_pitcher_base	90.0	99.0	**100.0**	**100.0**	99.6	**100.0**
021_bleach_cleanser	70.9	73.8	75.9	**89.8**	79.7	80.7
024_bowl	30.5	37.7	18.5	68.0	**92.7**	90.0
025_mug	40.7	61.5	77.5	89.0	**100.0**	98.0
035_power_drill	63.5	78.5	97.4	95.9	**99.3**	**99.3**
036_wood_block	27.7	59.5	**87.6**	58.7	54.7	58.7
037_scissors	17.1	3.9	**71.8**	62.4	33.3	70.7
040_large_marker	4.8	7.4	23.3	18.8	82.7	**87.3**
051_large_clamp	25.6	69.8	87.6	**95.4**	90.7	94.7
052_extra_large_clamp	8.8	90.0	98.0	95.6	62.0	**99.3**
061_foam_brick	34.7	71.9	**99.3**	87.2	72.0	70.7
Mean	39.0	60.1	80.5	81.4	79.2	**83.2**

Table 4. Detailed results on YCB-V [22] w.r.t. ADD(-S). We highlight the best results in **bold**.

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

Figure 1. **Qualitative results on T-LESS [8]**: (a, c) The original images and (b, d) MRC-Net object pose predictions. The object's 3D model is projected with estimated 6D pose and overlaid on original images with distinct colors. Mask RCNN [7, 10] detection is used. Best viewed when zoomed in.

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

Figure 2. **Qualitative results on YCB-V [22]**: (a, c) The original images and (b, d) MRC-Net object pose predictions. Mask RCNN [7, 10] detection is used. Best viewed when zoomed in.

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

Figure 3. **Qualitative results on LM-O [1]**: (a, c) The original images and (b, d) MRC-Net object pose predictions. Mask RCNN detection [7, 10] is used. Best viewed when zoomed in.

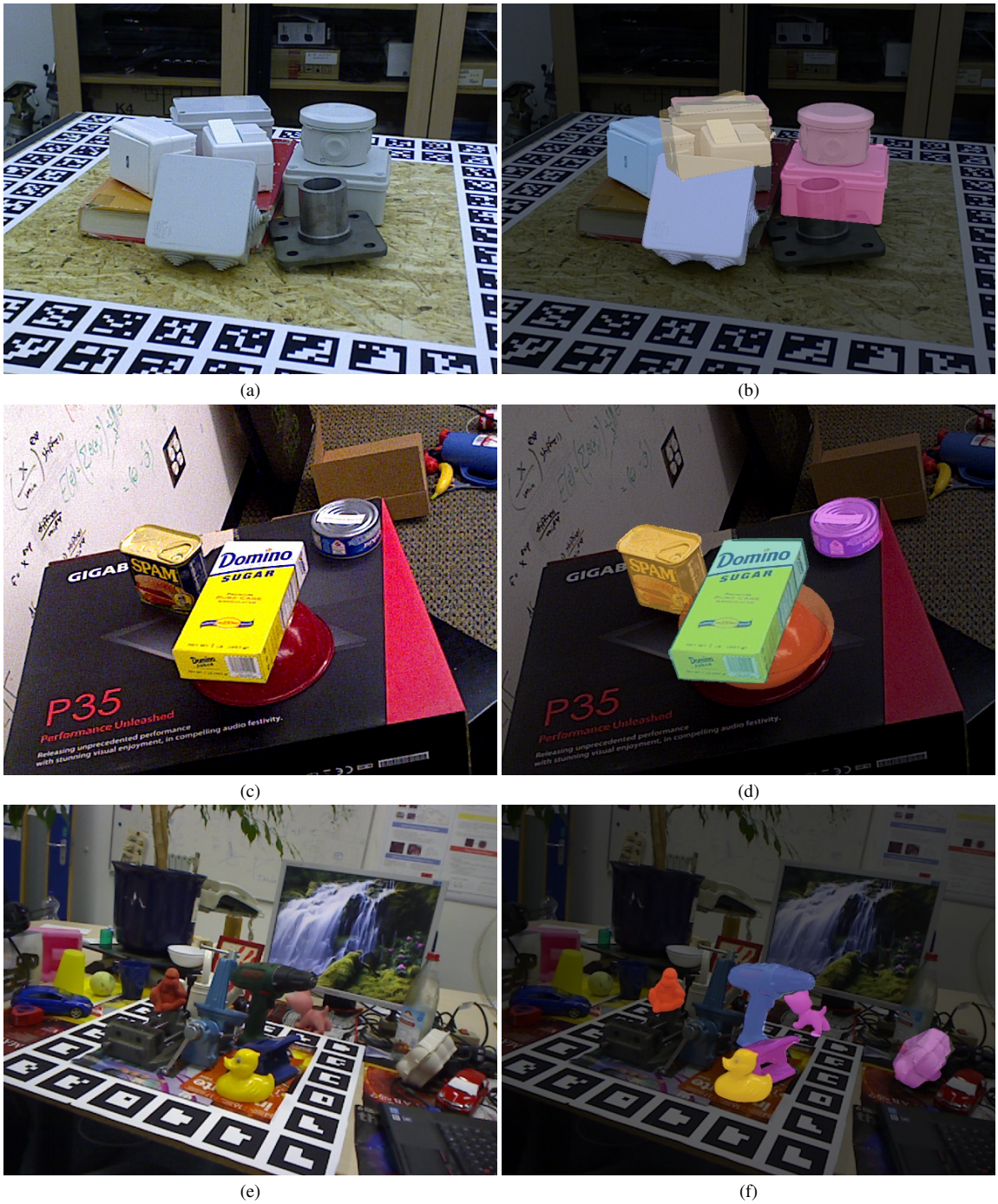

Figure 4. **Failure Examples**: (a), (c), and (e) show the original images. (b), (d), and (f) represent MRC-net predictions. Observe the flipped object pose induced by heavy occlusion in the center in (b), the upside-down red bowl in (d), and the inaccurately rotated eggbox in (f).

References

[1] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *European Conference on Computer Vision*, pages 536–551. Springer, 2014. 7

[2] Dingding Cai, Janne Heikkilä, and Esa Rahtu. SC6D: Symmetry-agnostic and correspondence-free 6d object pose estimation. In *International Conference on 3D Vision*, pages 536–546. IEEE, 2022. 1

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017. 1

[4] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. 1

[5] Krzysztof M Gorski, Eric Hivon, Anthony J Banday, Benjamin D Wandelt, Frode K Hansen, Mstvos Reinecke, and Matthia Bartelmann. HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal*, 622 (2):759, 2005. 1, 2

[6] Rasmus Laurvig Haugaard and Anders Glent Buch. Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6749–6758, 2022. 1

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2961–2969, 2017. 3, 4, 5, 6, 7

[8] Tomáš Hodan, Pavel Haluza, Štepán Obdržálek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: An RGB-D dataset for 6d pose estimation of texture-less objects. In *IEEE Winter Conference on Applications of Computer Vision*, pages 880–888. IEEE, 2017. 3, 5

[9] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6d object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3385–3394, 2019. 4

[10] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. CosyPose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020. 1, 2, 3, 4, 5, 6, 7

[11] Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. MegaPose: 6d pose estimation of novel objects via render & compare. In *Proceedings of the 6th Conference on Robot Learning*, 2022. 2

[12] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 683–698, 2018. 1

[13] Zhigang Li, Gu Wang, and Xiangyang Ji. CDPN: coordinates-based disentangled pose network for real-time rgb-based 6-DoF object pose estimation. In *IEEE International Conference on Computer Vision*, pages 7677–7686, 2019. 1, 2, 3, 4

[14] Ruyi Lian and Tae-Kyun Kim. CheckerPose: Progressive dense keypoint localization for object pose estimation with graph neural network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 14022–14033, 2023. 3, 4

[15] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. In *International Conference on Learning Representations*, 2018. 1

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28, 2015. 3, 4

[17] Yongzhi Su, Mahdi Saleh, Torben Fetzer, Jason Rambach, Nassir Navab, Benjamin Busam, Didier Stricker, and Federico Tombari. ZebraPose: Coarse to fine surface encoding for 6dof object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6738–6748, 2022. 3, 4

[18] Martin Sundermeyer, Maximilian Durner, En Yen Puang, Zoltan-Csaba Marton, Narunas Vaskevicius, Kai O Arras, and Rudolph Triebel. Multi-path learning for object pose estimation across domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 13916–13925, 2020. 1

[19] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9627–9636, 2019. 3, 4

[20] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. GDR-Net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 16611–16621, 2021. 3, 4

[21] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision*, pages 3–19, 2018. 1

[22] 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*, 2018. 3, 4, 6

[23] Anna Yershova, Swati Jain, Steven M Lavalle, and Julie C Mitchell. Generating uniform incremental grids on SO(3) using the Hopf fibration. *The International Journal of Robotics Research*, 29(7):801–812, 2010. 1

[24] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *European Conference on Computer Vision*, 2020. 2