Supplementary Material for
HiPose: Hierarchical Binary Surface Encoding and Correspondence Pruning for RGB-D 6DoF Object Pose Estimation

1. Details of Network

Figure 1 illustrates the architecture of HiPose. The network comprises two branches, namely the RGB branch and the point cloud branch. In the pre-stage of the RGB branch, a cropped image with dimensions of $3\times H\times W$ is transformed into RGB embedding with dimensions of $128\times H/4\times W/4$. Here, H and W represent the height and width of the input cropped RGB image, respectively, and by default, both are set to 256. On the other hand, the point cloud branch maps an input with 9 channels, consisting of point coordinates, color, and normal information, to a feature. The parameter $npts$ is set to 2730, which denotes the number of randomly sampled valid input point clouds.

Each of these branches is constructed with interconnected encoders and decoders, serving the fundamental purpose of feature extraction, feature transformation, and feature fusion.

The process of feature extraction aims to extract high-level features and adjust the channel dimension. In accordance with FFB6D [3], RandLA-Net [5] is employed to handle point cloud features. Furthermore, pre-trained ConvNeXt-B [6] and PSPNet [8] models are incorporated into the encoder and decoder blocks.

Feature transformation refers to the conversion between the features of the RGB branch and the point cloud branch, facilitated through coordinate correspondence. Specifically, as demonstrated in Figure 2, the point cloud branch feature can be generated by aggregating features from the nearest features in the RGB branch. Likewise, the RGB branch feature can be generated by interpolating the feature from the point cloud branch. This enables bidirectional transformation between the features of the RGB branch and the point cloud branch.

The process of feature fusion is executed using a Convolutional Neural Network (CNN). The new RGB feature is generated by concatenating the RGB feature with the transformed RGB feature, and the same procedure is applied to the depth feature. Further details regarding the feature fusion process can be observed in Figure 3.

Finally, a straightforward convolution-based head is employed to predict the visible mask and code for the selected $npts$ points.

# points	4	6	8	10	20	30
500 iterations	88.7	89.0	89.0	**89.1**	89	88.7
1000 iterations	89.0	**89.1**	88.9	**89.1**	**89.1**	88.8
1500 iterations	89.0	**89.1**	**89.1**	**89.1**	88.9	88.8

Table 1. **Test RANSAC+Kabsch parameters on LM-O [1].** We tune the number of correspondences in each RANSAC iteration and the number of RANSAC iterations with a maximum correspondence points-pair distance of 2cm. The results are presented with average recall of ADD(-S) in %. According to the table, using 10 correspondences in each RANSAC iteration yields the best results. However, the results achievable with RANSAC+Kabsch are inferior to those obtained with our hierarchical approach.

2. Details of Open3D RANSAC+Kabsch

We use the $registration\_ransac\_based\_on\_correspondence$ function in Open3D[10] to solve the object pose with the given correspondence. We tuned the number of correspondences in each RANSAC iteration and the number of RANSAC iterations. The results achievable with RANSAC+Kabsch are inferior to those obtained with our hierarchical approach, as showed in Table. 1.

3. Impact of ICP

The Iterative Closest Point algorithm (ICP) is commonly employed as a refinement strategy, leveraging depth information to align the estimated pose. We assess the impact of ICP on both HiPose and ZebraPose, both of which are trained solely with pbr images in Table. 2. For HiPose, we provide a ground truth object mask to facilitate the application of ICP. Surprisingly, ICP fails to yield any enhancements and, in fact, worsens the outcome. In the case of ZebraPose, a substantial improvement in the result is observed. Nevertheless, once the pose achieves a satisfactory level of accuracy, such as employing RANSAC Kabsch (recall greater than 87%), the incorporation of ICP does not

Experiment Setup	ADD(-S) in %
ZebraPose (Trained only with pbr images)	63.5
ZebraPose (pbr) + ICP	83.9
ZebraPose (pbr) + RANSAC Kabsch	87.0
ZebraPose (pbr) + RANSAC Kabsch + ICP	87.0
HiPose (ours)	**89.6**
HiPose + ICP refinement (with ground truth object mask)	89.3

Table 2. **Evaluate the impact of ICP.** We assess the impact of ICP on both HiPose and ZebraPose, both of which are trained solely with pbr images. We observed that once the pose achieves a satisfactory level of accuracy, the incorporation of ICP does not lead to betterment.

lead to betterment. This circumstance may be attributed to insufficient accuracy in the depth map.

4. Impact of noisy depth

Experiment Setup	ADD(-S) in %
HiPose (ours)	**89.6**
Depth with Zero Mean Gaussian Noise with Sigma 0.01	89.0
Random drop 20% points in Depth Map	89.5

Table 3. **Evaluate the impact of noisy depth.** When introducing noise or randomly omitting data points in the depth map, HiPose still performs admirably under such circumstances.

During training, we augmented the depth maps with Gaussian noise and randomly dropped pixels, to make the network less sensitive to the noise. Coincidentally, the 3 evaluated datasets are captured with different depth sensors, showing that HiPose is robust to different noise levels. We perform additional experiments in Table. 3, showing that HiPose is quite robust to missing measurements in the depth map. However, inaccurate measurements do slightly affect performance.

5. Details of YCB-V results

We summarized the per-object results on the YCB-V dataset [7] in Table 4. As presented in the table, we outperform other approaches on most test objects.

6. Qualitative Results

We present quantitative results on the LM-O [1], YCB-V [7], and T-LESS [4] datasets in Figure 4, Figure 5, and Figure 6, respectively. We rendered the object into the image using the estimated pose. It is clear to see that the contour of the rendered object aligns seamlessly with the real object in the image, demonstrating the accuracy of our estimated pose. Furthermore, it is evident that our proposed HiPose performs well with texture-less objects and can handle occlusion effectively.

References

[1] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3364–3372, 2016. 1, 2, 5

[2] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11632–11641, 2020. 3

[3] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3003–3013, 2021. 1, 3

[4] 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 *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 880–888. IEEE, 2017. 2, 7

[5] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11108–11117, 2020. 1, 4

[6] Zhuang Liu, Hanzi Mao, Chaozheng Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, 2022. 1, 4

[7] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. 2018. 2, 6

[8] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 1, 4

[9] Jun Zhou, Kai Chen, Linlin Xu, Qi Dou, and Jing Qin. Deep fusion transformer network with weighted vector-wise keypoints voting for robust 6d object pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13967–13977, 2023. 3

[10] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. 1

Method	PVN3D [2]		FFB6D [3]		DFTr [9]		**Ours**	
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)
002_master_chef_can	96.0	80.5	96.3	80.6	**97.0**	**92.3**	96.4	86.2
003_cracker_box	96.1	94.8	96.3	94.6	95.9	93.9	**97.7**	**96.7**
004_sugar_box	97.4	96.3	97.6	96.6	97.1	95.5	**98.2**	**97.1**
005_tomato_soup_can	96.2	88.5	95.6	89.6	95.6	92.6	**97.0**	**95.1**
006_mustard_bottle	97.5	96.2	97.8	**97.0**	97.6	96.3	**98.4**	96.9
007_tuna_fish_can	96.0	89.3	96.8	88.9	97.3	94.5	**97.8**	**96.2**
008_pudding_box	97.1	95.7	97.1	94.6	97.4	95.7	**98.8**	**98.1**
009_gelatin_box	97.7	96.1	98.1	96.9	97.6	96.3	**98.9**	**97.8**
010_potted_meat_can	93.3	88.6	94.7	88.1	**95.9**	**92.1**	93.5	83.4
011_banana	96.6	93.7	97.2	94.9	97.1	95.0	**98.6**	**96.3**
019_pitcher_base	97.4	96.5	**97.6**	**96.9**	96.0	93.1	96.8	93.2
021_bleach_cleanser	96.0	93.2	96.8	94.8	96.8	**94.9**	**97.1**	94.0
024_bowl*	90.2	90.2	96.3	96.3	96.9	96.9	**98.0**	**98.0**
025_mug	97.6	95.4	97.3	94.2	97.6	94.9	**98.2**	**95.7**
035_power_drill	96.7	95.1	97.2	95.9	96.9	95.2	**98.3**	**97.4**
036_wood_block*	90.4	90.4	92.6	92.6	96.2	96.2	**97.0**	**97.0**
037_scissors	96.7	92.7	97.7	95.7	97.2	93.3	**98.3**	**96.8**
040_large_marker	96.7	91.8	96.6	89.1	96.9	92.7	**98.6**	**94.3**
051_large_clamp*	93.6	93.6	**96.8**	**96.8**	96.3	96.3	95.9	95.9
052_extra_large_clamp*	88.4	88.4	96.0	96.0	**96.4**	**96.4**	95.6	95.6
061_foam_brick*	96.8	96.8	97.3	97.3	97.3	97.3	**98.6**	**98.6**
mean	95.5	91.8	96.6	92.7	96.7	94.4	**97.5**	**95.3**

Table 4. **Comparison with State of the Art on YCB-V**. We report the Average Recall w.r.t AUC of ADD(-S) and AUC of ADD-S in % and compare with state of the art. (*) denotes symmetric objects.

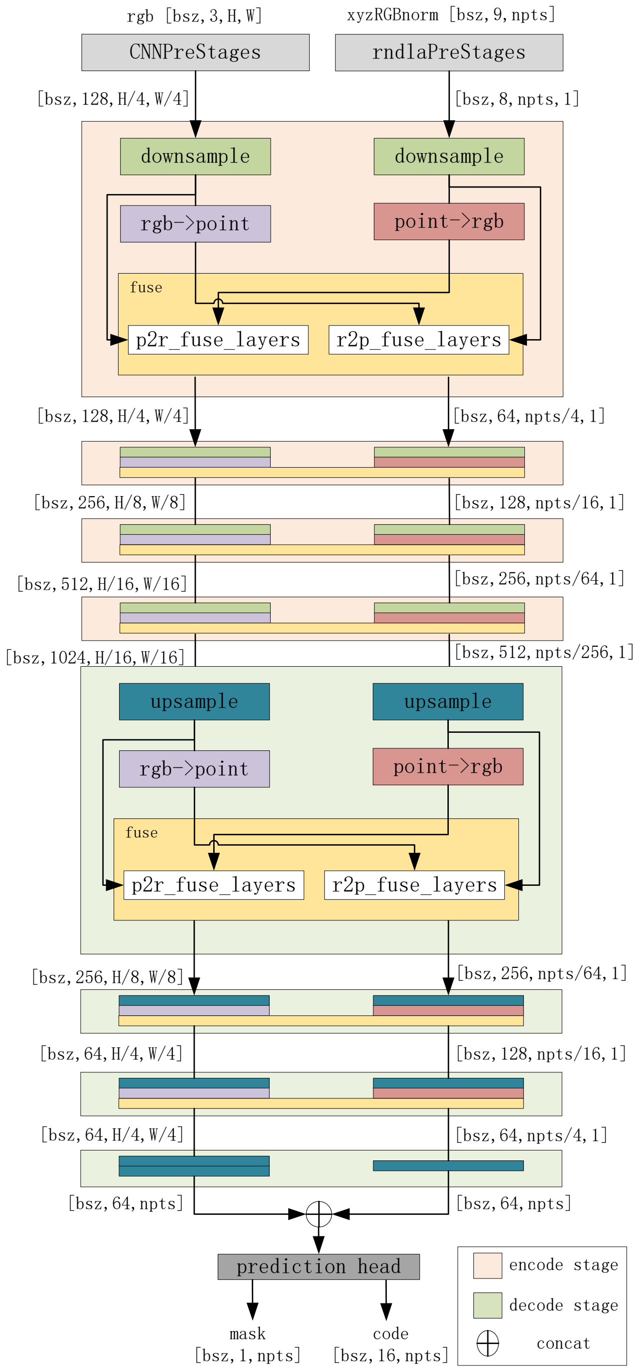

Figure 1. **Network Architecture :** The network comprises four encoder blocks and four decoder blocks. Each block performs upsampling or downsampling of the input, processes the RGB and point features, and subsequently merges them except the last decoder block. In the RGB image branch, we employ ConvNeXt blocks [6] as the encoders and PSPNet blocks [8] as the decoders. As for the point cloud branch, we utilize modules derived from Randla [5]. Here, 'bsz' refers to the batch size, 'npts' denotes the number of points, and 'H/W' represents the height and width of the image.

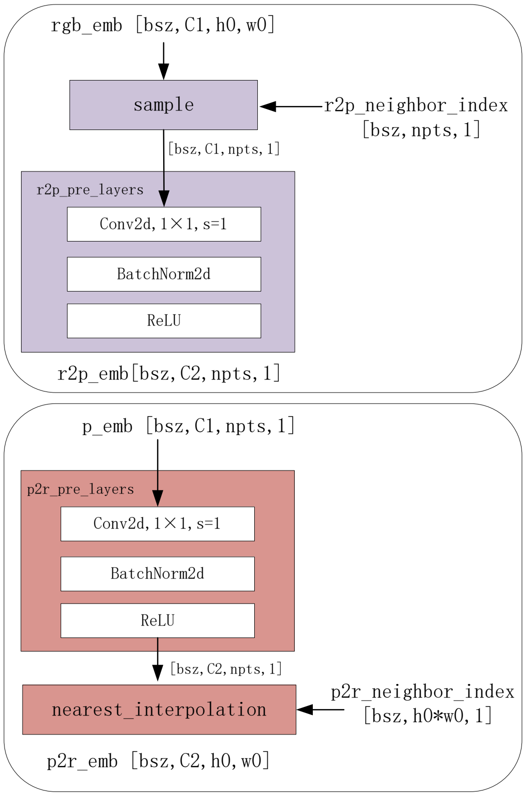

Figure 2. The upper block represents the conversion of RGB features to point features, denoted by $r2p\_emb$, while the bottom block illustrates the conversion of point features to RGB features, denoted by $p2r\_emb$. $r2p\_neighbor\_index$ indicates the index of the closest pixel for each point feature, similar for $p2r\_neighbor\_index$.

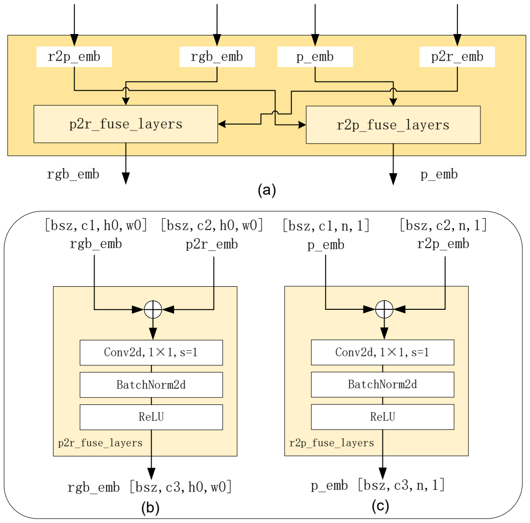

Figure 3. (a) The feature flow of the fuse block. (b) This block serves the purpose of fusing RGB feature rgb_emb and point-to-RGB feature $p2r\_emb$. (c) This block is responsible for fusing point feature p_emb and RGB-to-point feature $r2p\_emb$.

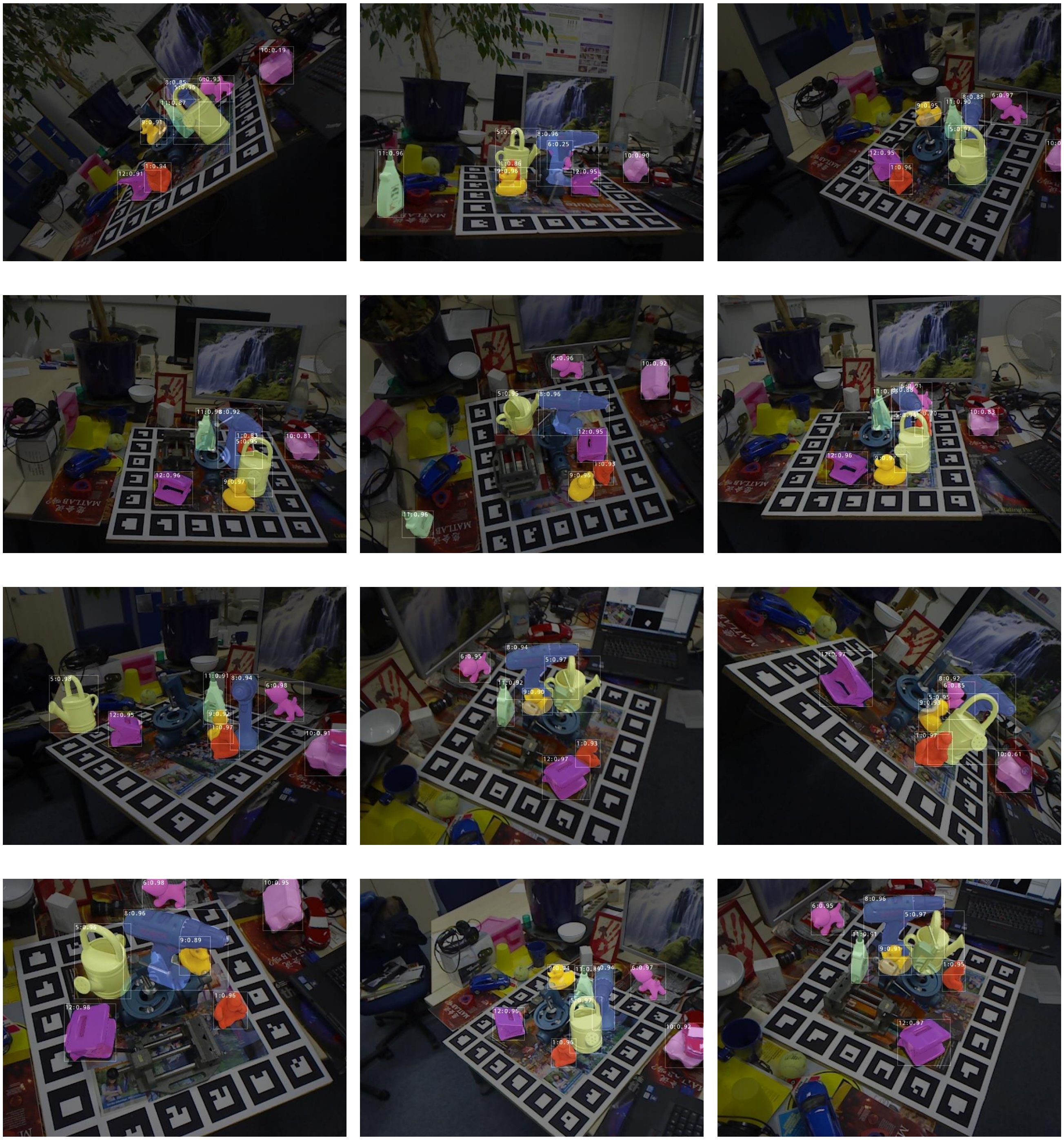

Figure 4. Qualitative Results on LM-O [1].

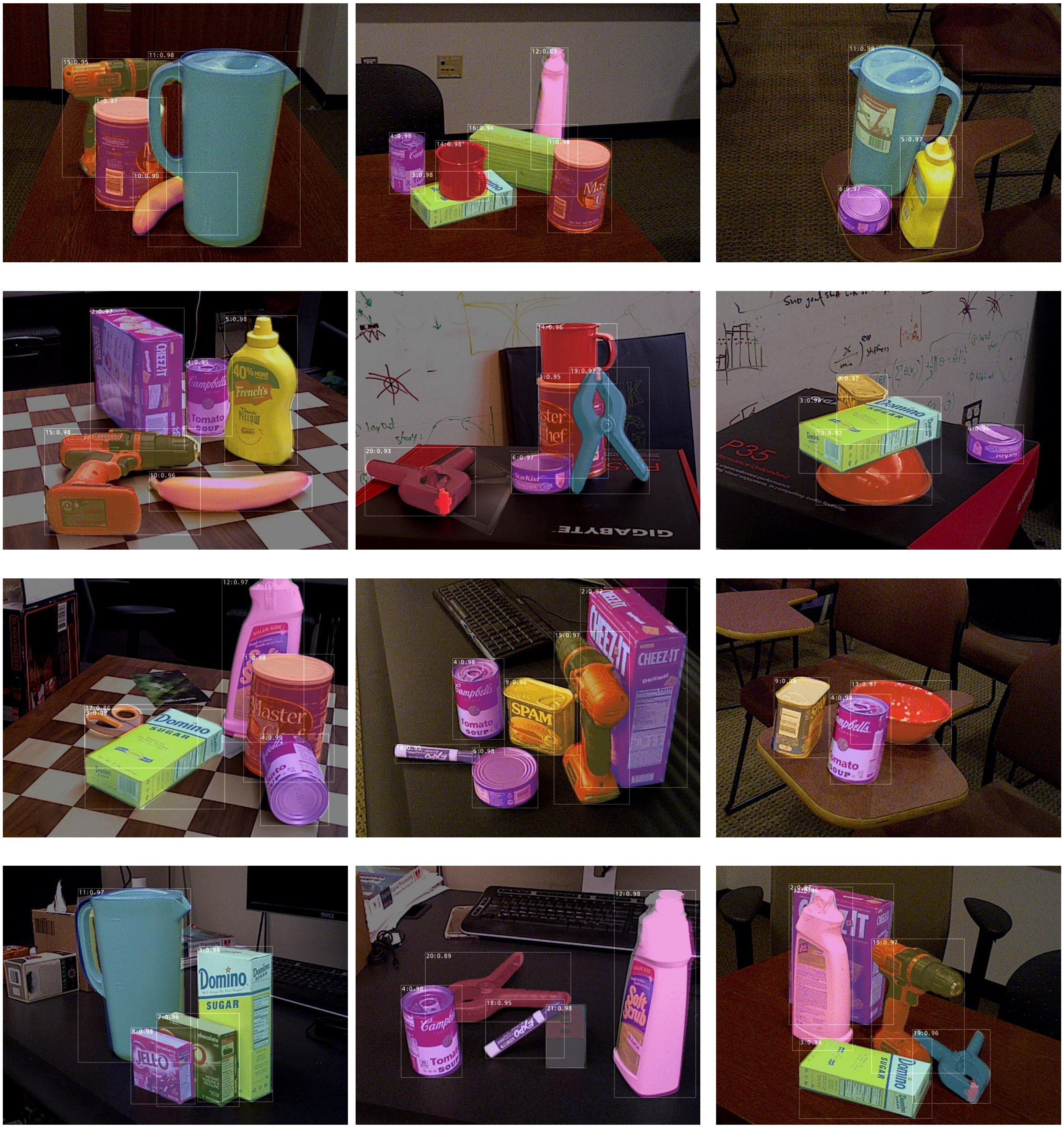

Figure 5. Qualitative Results on YCB-V [7].

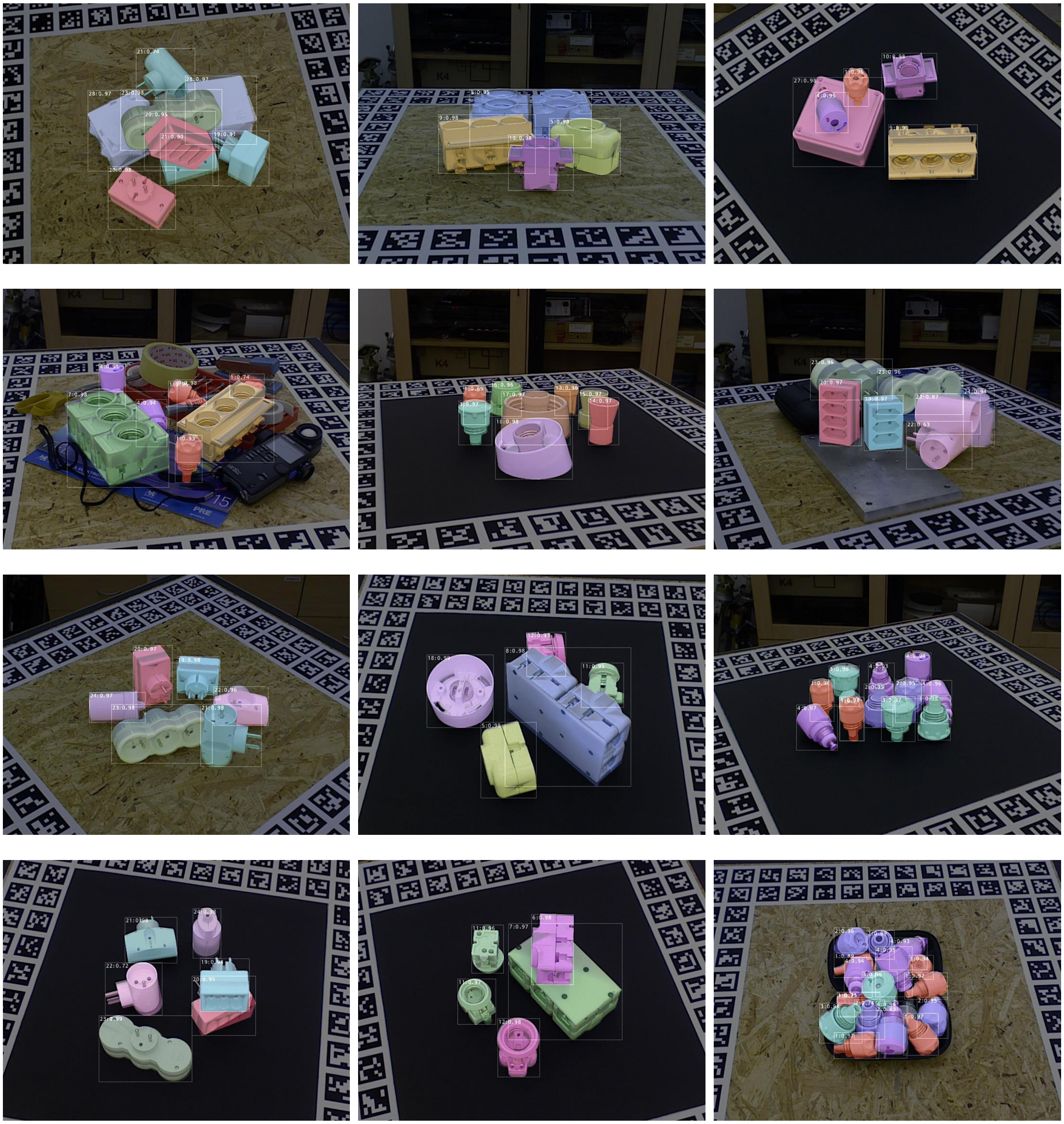

Figure 6. Qualitative Results on T-LESS [4].