Supplementary Material of
CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation

Zhigang Li Gu Wang Xiangyang Ji
Tsinghua University
Beijing, China
{lzg15, wangg16}@mails.tsinghua.edu.cn xyji@tsinghua.edu.cn

In this supplementary document, we first provide the detailed analysis on why the translation performance varies dramatically among objects when the pose is solved from coordinates. It impels us to develop a novel disentangled network to solve translation and rotation independently in the main paper. Then, we present the implementation details. Finally, we evaluate the influence of RANSAC iterations and provide more detailed results on the LINEMOD dataset.

A. Detailed Analysis on Translation Solved from Coordinates

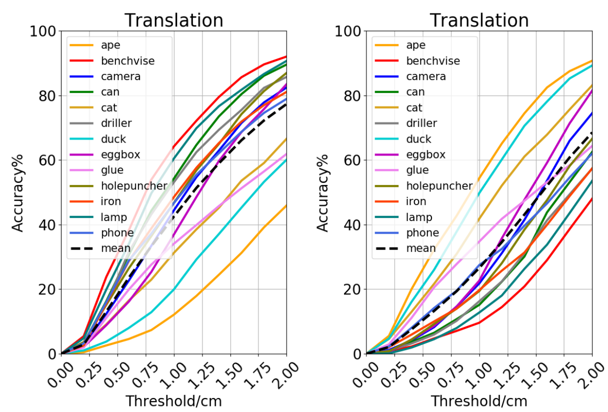
As mentioned in the main paper, the pose solved from coordinates shows diverse performance on ADD metric across different object catergories, which is mainly caused by the unbalanced translation performance (Fig. 3(a) in main paper). It shows that the method lacks robustness and fails in some cases.

Here we analyze the main cause of the unbalanced performance regarding translation estimation. The camera imaging process can be described as a full perspective camera model in Eq. 1.

$$w[u\ v\ 1]^T = \mathbb{K}[\mathbf{R}\ \mathbf{T}][x\ y\ z\ 1]^T, s.t., \mathbf{R}^T\mathbf{R} = \mathbf{I} \qquad (1)$$

where $\mathbb{K}$, $\mathbf{R}$ and $\mathbf{T}$ are the camera intrinsic parameters, rotation matrix, translation vector. w is a scale factor.

After building the 2D-3D correspondences from the predicted coordinates-confidence map, the pose can be solved by minimizing the 2D projection error via PnP (and maximizing the inliers via RANSAC). The coordinates $[x, y, z]$, pixels $[u, v]$ and the PnP+RANSAC algorithm can affect the translation $\mathbf{T}$. For PnP+RANSAC, evidently, the influence should be consistent and the same for all objects. In terms of the influence from 2D pixels, we erode the confidence map to maintain high-confidence object pixels and ensure no background pixels are selected. However, the problem still exists. So, the crux lies in the coordinates $[x, y, z]$. We analyze each translation component $\mathbf{T}_x$, $\mathbf{T}_y$ and $\mathbf{T}_z$. See the Fig. 3(b) in main paper, the inaccuracy mainly comes from $\mathbf{T}_z$, i.e. the depth between object and camera. Since the depth is affected by the size ratio of the object in image to the 3D object, the main cause of the problem probably lies in the 'scale' error of the predicted object coordinates. To verify it, we multiply the predicted coordinates with a scalar coefficient f and solve the pose. See the results in Fig. 1, the translation accuracy of 'ape' amazingly increases from 45.9% to 90.76% on threshold $2cm$ when $f = 1.03$. However, the accuracy of 'benchvise' drops from 92.05% to 48.01%. So, the real cause is the inaccurate coordinates 'scale'. Different objects own different scale errors δ_{scale}, which results in the dramatical diverse translation perfor-



(a) f = 1 (b) f = 1.03

Figure 1: (a) Translation solved from coordinates without scaling. (b) Translation solved from $f\cdot$ coordinates, where f is a coefficient to compensate the scale error. The results indicate that the unbalanced translation performance is caused by the scale error of coordinates.

Maximum Iters	5	10	20	50	100	> 100
5cm 5°	93.68	94.12	94.30	94.31	94.31	~ 94.31

Table 1: Evaluation on RANSAC iterations.

mance across different object catergories.

To solve the problem caused by the scale errors δ_{scale}. A direct way is to introduce a coefficient f for each object to compensate δ_{scale} after training. However, the prerequisite for this method is that training samples share the same δ_{scale} with test samples. When the training data and test data come from different sources (e.g. synthetic training data vs. real test data), the δ_{scale} can be different. Moreover, finding a proper f for each object is quite time-consuming and tedious. Since the scale error δ_{scale} only affects translation, we propose to disentangle the pose estimation to solve this problem by indirectly solving rotation from coordinates via PnP while directly regressing translation from image (see the main paper).

B. Implementation Details

Network Architecture. In CDPN, we use ResNet34 as our backbone net. Then, we define $deconv1/conv2/conv3$ as a up-scaling block, including a deconvolutional layer (kernel 3×3, stride 2, channel 256, relu, bn) and two convolutional layers (kernel 3×3, stride 1, channel 256, relu, bn). The rotation head is built by stacking three up-scaling blocks with an additional output convolutional layer (kernel 1×1, stride 1, channel 4). For the translation head, it includes six $conv$ (kernel 3×3, stride 1, channel 256, relu, bn) layers and subsequent three fully-connected layers (4096-4096-3).

Training. Our approach was implemented in the Pytorch framework [4]. We trained all categories using one network. The parameters in *Dynamic Zoom In* were set as follows: $\alpha = \beta = \gamma = 0.25$, $\rho = 1.5$ and $\sigma_1 = \sigma_2 = \sigma_3 = 1$. For *Masked Coordinates-Confidence Loss*, we set $\alpha = \beta = 1$. When building 2D-3D correspondences, we used 0.5 as the threshold of the confidence map to extract the 2D pixels. For *Scale-invariant Translation Estimation*, we first converted the unit of 3D coordinates to meter and then set $\gamma_1 = \gamma_2 = \gamma_3 = 1$. During training, the initial learning rate was 1×10^{-4} and the batch size was 6. We used RMSProp with alpha 0.99 and epsilon 1×10^{-8} to optimize the network. The model was trained for 160 epochs in total and the learning rate was divided by 10 every 50 epochs.

C. RANSAC Iterations

Here, we evaluate the influence of the number of RANSAC iterations (Table 1). The results show that a few iterations (e.g. 20) are enough for our approach to achieve highly accurate pose estimation, which is crucial for building a fast real-time system considering that RANSAC is quite time-consuming when the number of iterations is large.

D. Detailed Results on the LINEMOD Dataset

Table 2, 3, 4 show the detailed results of the comparison to the state-of-the-art RGB-only methods on LINEMOD dataset. It is worth noting that even without refinement, our approach still outperforms those methods refined with depth and ICP.

E. More Qualitative Results

See more qualitative 6-DoF pose estimation results in Fig. 2 3 4.

References

[1] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir 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 (CVPR)*, 2017.

[3] Apurv Nigam, Adrian Penate-Sanchez, and Lourdes Agapito. Detect globally, label locally: Learning accurate 6-dof object pose estimation by joint segmentation and coordinate regression. *IEEE Robotics and Automation Letters (RAL)*, 2018.

[4] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[5] Mahdi Rad and Vincent Lepetit. BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[6] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of The European Conference on Computer Vision (ECCV)*, 2018.

[7] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-Time Seamless Single Shot 6D Object Pose Prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

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

	w/o Refinement								w/ Refinement			
Method	BB8 [5]	YOLO6D [7]	PoseCNN [8]	SSD6D [2]	AAE [6]	Brachmann [1]	Nigam [3]	Ours	BB8 [5]	SSD6D [2]	Brachmann [1]	AAE [6]
ape	-	-	7.0	-	-	-	47.7	**81.62**	80.2	-	34.4	-
benchvise	-	-	13.6	-	-	-	37.9	**97.87**	81.5	-	40.6	-
camera	-	-	20.4	-	-	-	31.5	**98.43**	60.0	-	30.5	-
can	-	-	24.9	-	-	-	48.5	**99.11**	76.8	-	48.4	-
cat	-	-	25.1	-	-	-	37.4	**96.01**	79.9	-	34.6	-
driller	-	-	18.2	-	-	-	-	**96.63**	69.6	-	54.5	-
duck	-	-	18.2	-	-	-	52.8	**90.14**	53.2	-	22.0	-
eggbox	-	-	33.3	-	-	-	-	**98.78**	81.3	-	57.1	-
glue	-	-	19.5	-	-	-	-	**82.82**	54.0	-	23.6	-
holepuncher	-	-	15.9	-	-	-	-	**98.38**	73.1	-	47.3	-
iron	-	-	13.1	-	-	-	41.6	**93.56**	61.1	-	58.7	-
lamp	-	-	24.4	-	-	-	51.9	**98.85**	67.5	-	49.3	-
phone	-	-	19.3	-	-	-	-	**93.77**	58.6	-	26.8	-
Average	-	-	19.4	-	-	-	43.7	**94.31**	69.0	-	40.6	-

Table 2: Comparison with state-of-the-art RGB-only methods on **5cm 5°**.

	w/o Refinement								w/ Refinement			
Method	BB8 [5]	YOLO6D [7]	PoseCNN [8]	SSD6D [2]	AAE [6]	Brachmann [1]	Nigam [3]	Ours	BB8 [5]	SSD6D [2]	Brachmann [1]	AAE [6]
ape	27.9	21.62	27.8	0.00	3.96	-	-	64.38	40.4	**65**	33.2	20.55
benchvise	62.0	81.80	68.9	0.18	20.92	-	-	**97.77**	91.8	80	64.8	64.25
camera	42.1	36.57	47.5	0.41	30.47	-	-	**91.67**	55.7	78	38.4	63.20
can	48.1	68.80	71.4	1.35	35.87	-	-	**95.87**	64.1	86	62.9	76.09
cat	45.2	41.82	56.7	0.51	17.90	-	-	**83.83**	62.6	70	42.7	72.01
driller	58.6	63.51	65.4	2.58	23.99	-	-	**96.23**	74.4	73	61.9	41.58
duck	32.8	27.23	42.8	0.00	4.86	-	-	**66.76**	44.3	66	30.2	32.38
eggbox	40.0	69.58	98.3	8.90	81.01	-	-	99.72	57.8	**100**	49.9	98.64
glue	27.0	80.02	95.6	0.00	45.49	-	-	99.61	41.2	**100**	31.2	96.39
holepuncher	42.4	42.63	50.9	0.30	17.60	-	-	**85.82**	67.2	49	52.8	49.88
iron	67.0	74.97	65.6	8.86	32.03	-	-	**97.85**	84.7	78	80.0	63.11
lamp	39.9	71.11	70.3	8.20	60.47	-	-	**97.89**	74.5	73	67.0	91.69
phone	35.2	47.74	54.6	0.18	33.79	-	-	**90.75**	54.0	79	38.1	70.96
Average	43.6	55.95	62.7	2.42	31.41	32.3	-	**89.86**	62.7	79	50.2	64.67

Table 3: Comparison with state-of-the-art RGB-only methods on **ADD**.

	w/o Refinement								w/ Refinement			
Method	BB8 [5]	YOLO6D [7]	PoseCNN [8]	SSD6D [2]	AAE [6]	Brachmann [1]	Nigam [3]	Ours	BB8 [5]	SSD6D [2]	Brachmann [1]	AAE [6]
ape	95.3	92.10	83.0	-	-	-	-	**96.86**	96.6	-	85.2	-
benchvise	80.0	95.06	50.0	-	-	-	-	**98.35**	90.1	-	67.9	-
camera	80.9	93.24	71.9	-	-	-	-	**98.73**	86.0	-	58.7	-
can	84.1	97.44	69.8	-	-	-	-	**99.41**	91.2	-	70.8	-
cat	97.0	97.41	92.0	-	-	-	-	**99.80**	98.8	-	84.2	-
driller	74.1	79.41	43.6	-	-	-	-	**95.34**	80.9	-	73.9	-
duck	81.2	94.65	91.8	-	-	-	-	**98.59**	92.2	-	73.1	-
eggbox	87.9	90.33	91.1	-	-	-	-	**98.97**	91.0	-	83.1	-
glue	89.0	96.53	88.0	-	-	-	-	**99.23**	92.3	-	74.2	-
holepuncher	90.5	92.86	82.1	-	-	-	-	**99.71**	95.3	-	78.9	-
iron	78.9	82.94	41.8	-	-	-	-	**97.24**	84.8	-	83.6	-
lamp	74.4	76.87	48.4	-	-	-	-	**95.49**	75.8	-	64.0	-
phone	77.6	86.07	58.8	-	-	-	-	**97.64**	85.3	-	60.6	-
Average	83.9	90.37	70.2	-	-	69.5	-	**98.10**	89.3	-	73.7	-

Table 4: Comparison with state-of-the-art RGB-only methods on **Proj. 2D**.

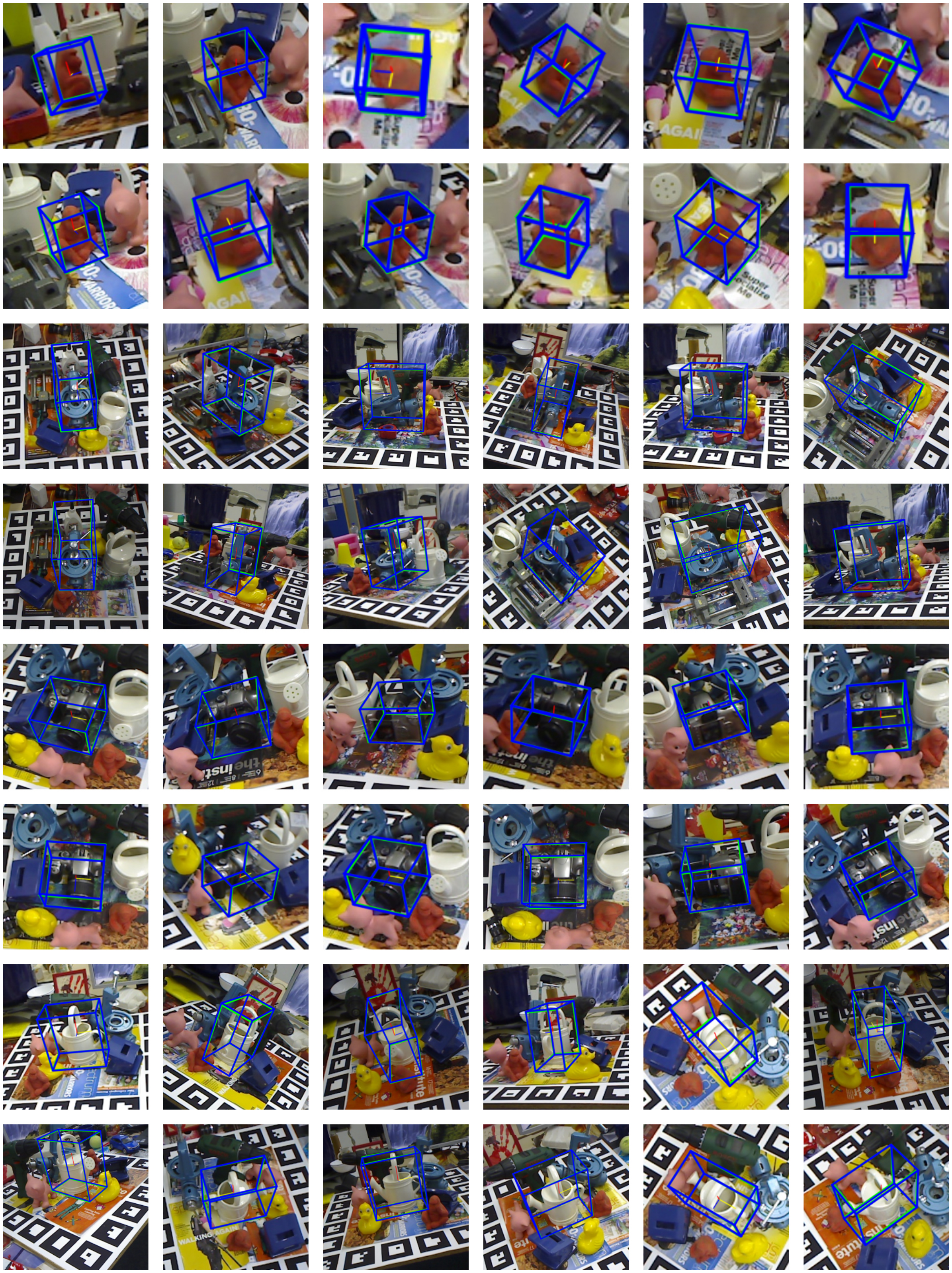


Figure 2: Qualitative results on the LINEMOD dataset. The green 3D bounding boxes represent the ground truth while the blue ones represent our predictions.

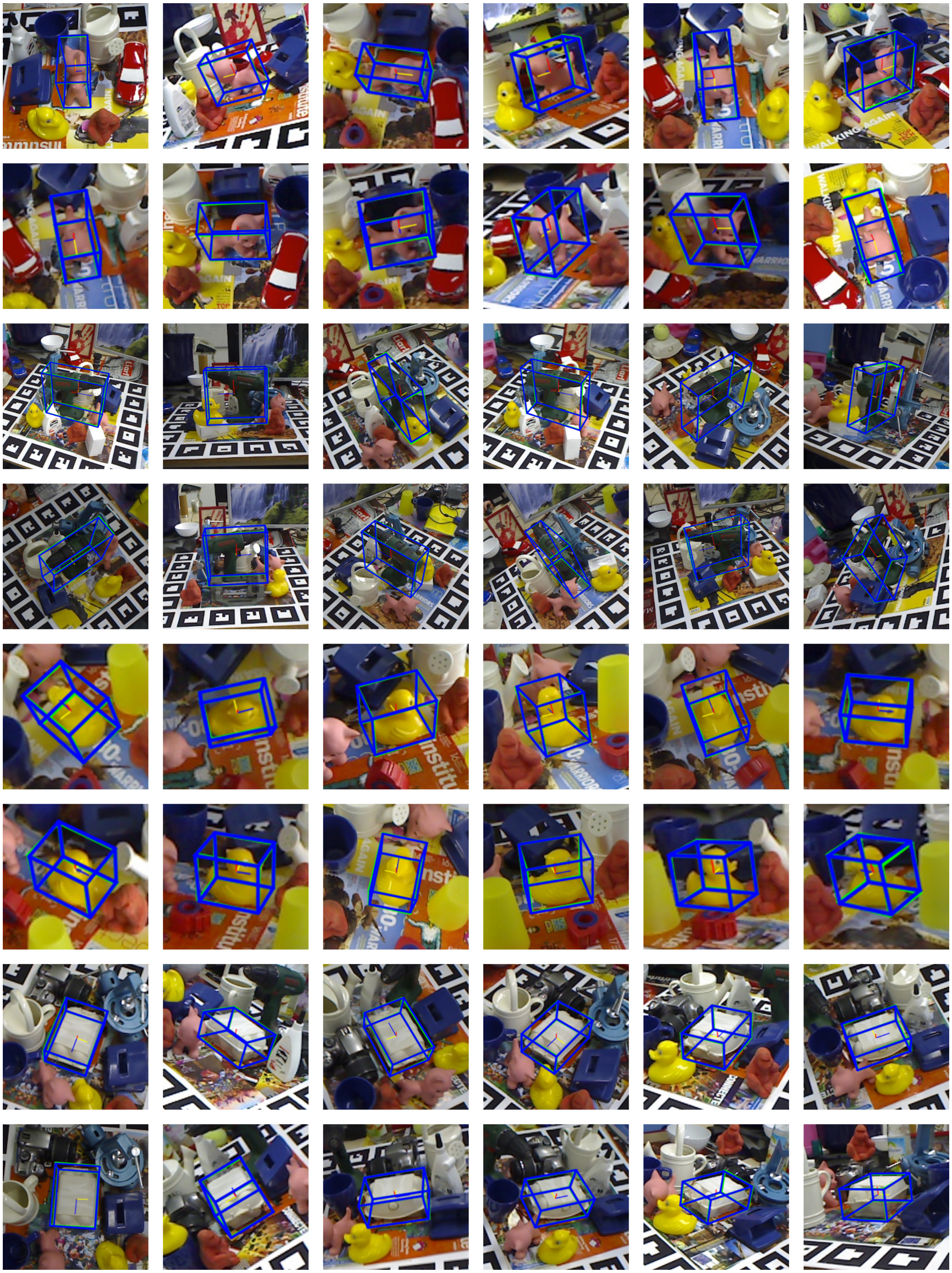


Figure 3: Qualitative results on the LINEMOD dataset. The green 3D bounding boxes represent the ground truth while the blue ones represent our predictions.

Figure 4: Qualitative results on the LINEMOD dataset. The green 3D bounding boxes represent the ground truth while the blue ones represent our predictions.