Appendix for Uni6D: A Unified CNN Framework without Projection Breakdown for 6D Pose Estimation

1. Implementation details

1.1. The details of the positional encoding.

PE is implemented using equation 1 and the details will be added in the final version.

$$\begin{aligned}
&PE(x, y, 2i) = sin(x/10000^{(4i/D)})\\
&PE(x, y, 2i+1) = cos(x/10000^{(4i/D)})\\
&PE(x, y, 2j+D/2) = sin(y/10000^{(4j/D)})\\
&PE(x, y, 2j+1+D/2) = cos(y/10000^{(4j/D)}),
\end{aligned} \tag{1}$$

where (x, y) is a point in 2d space, i, j is an integer in $[0, D/4)$, D is the size of the channel dimension.

1.2. The details of the pre-trained weight.

We use the ImageNet pre-trained weight, and the first convolutional layer is initialized with the kaiming uniform. **For YCB dataset**:

- Backbone: ResNet50 + FPN;
- Input data: RGB-D+UV+PE+XY+NRM, rotation matrices are represented by quaternions, other settings are same with PVN3D [1];
- Data augmentation:
    1. multi-scale training: [320, 400, 480, 600, 720] (max size is 900);
    2. background replacing: replace the background of the rendered data with the real image background;
    3. random crop: 0.3 probability, need to keep all objects;
- Training:
    1. Pretrained: ImageNet;
    2. Schedule: 40epoch, MultiStepLR with [15, 25, 35] schedule and 0.1× decay ;
    3. Optimizer: SGD, momentum 0.9, weight_deacy 0.0001, warm-up 4 epoch;
- Loss function:
    1. $Loss = \alpha Addloss + RPNloss + bboxloss + clsloss + maskloss + abcheadloss$;
    2. α is changed in training: 1-15 epoch is 1, 16-25 epoch is 5, 26-35 epoch is 10 and 36-40 epoch is 20;

For Linemode dataset:

- Backbone: ResNet50 + FPN;
- Input data: RGB-D+UV+PE+XY+NRM, rotation matrices are represented by quaternions, other settings are same with PVN3D [1], except using camera intrinsics for real data to render data;
- Data augmentation:
    1. multi-scale training: [320, 400, 480, 600, 720] (max size is 900);
    2. background replacing: replace the background of the rendered data with the real image background;
    3. random crop: 0.3 probability, need to keep all objects;
    4. random erase: 0.1 probability
- Training:
    1. Pretrained: ImageNet;
    2. Schedule: 40epoch, MultiStepLR with [15, 25, 35] schedule and 0.1× decay ;
    3. Optimizer: SGD, momentum 0.9, weight_deacy 0.0001, warm-up 4 epoch;
- Loss function:
    1. $Loss = \alpha Addloss + RPNloss + bboxloss + clsloss + maskloss + abcheadloss$;
    2. α is changed in training: 1-15 epoch is 1, 16-25 epoch is 5, 26-35 epoch is 10 and 36-40 epoch is 20;

2. Ablation Studies of abc Head

We provide results of more ablation studies for abc head on YCB dataset in Table 1. We combine the abc head with different UV input information to verify the effectiveness of it. We can observe that our abc head can improve the performance **without** UV and it can further improve the performance **with** different types of UV. These results demonstrate the effectiveness of abc head as an auxiliary training task.

	RGB-D	Plain UV	XY	PE	NRM
w/o	90.99/79.72	94.06/85.39	94.17/85.66	**93.54**/85.05	93.79/84.79
w	**91.13/80.89**	**94.49/86.46**	**94.33/86.90**	93.53/**86.09**	**93.89/84.96**

Table 1. Ablation study results of abc head, the format is ADDS/ADD(S).

3. Quantitative Results on the LineMOD Dataset

Experimental results of LineMOD dataset are reported in Table 2, our approach achieves 97.03% ADD-0.1d ACC with a succinct and straightforward pipeline compared with other methods. LineMOD is usually thought to be less challenging due to the varying lighting conditions, significant image noise and occlusions in YCB-Video Dataset.

4. Quantitative Results on the Occlusion LineMOD dataset

We follow the previous works [7,9] to train our model on the LineMOD dataset and only use this dataset for testing. Experimental results of LineMOD dataset are reported in Table 3, and we obtain 30.71 ADDS-0.1d AUC.

5. More Qualitative Results

We give more qualitative comparison results between our method and the SOTA method FFB6D [9] in Fig. 1 for YCB-Video dataset and Fig. 2 for LineMOD dataset. Moreover, **We strongly recommend readers to watch the video from https://youtu.be/6G_P282djw , which directly reflects the comparison results between our method and the FFB6D [9].** Compared with FFB6D, our method estimates the 6d pose more smoothly. Our method has better consistency between adjacent frames, less jitter, and more robust performance under severe occlusion conditions.

References

[1] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 3

[2] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. 2018. 3

[3] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4561–4570, 2019. 3

[4] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7678–7687, 2019. 3

[5] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1941–1950, 2019. 3

[6] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 244–253, 2018. 3

[7] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3343–3352, 2019. 2, 3

[8] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, and Ales Leonardis. G2l-net: Global to local network for real-time 6d pose estimation with embedding vector features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4233–4242, 2020. 3

[9] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 2, 3

[10] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134, 2018. 3

[11] P Kiru, Timothy Patten, and MV Pix2Pose. Pixel-wise coordinate regression of objects for 6d pose estimation. *Proceedings of the ICCV, Seoul, Korea*, pages 27–28, 2019. 3

[12] Yinlin Hu, Pascal Fua, Wei Wang, and Mathieu Salzmann. Single-stage 6d object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2930–2939, 2020. 3

[13] Chen Song, Jiaru Song, and Qixing Huang. Hybridpose: 6d object pose estimation under hybrid representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 431–440, 2020. 3

Table 2. Evaluation results (ADD-0.1d ACC) on the LineMOD dataset. Symmetric objects are denoted in bold.

	PoseCNN [2]	PVNet [3]	CDPN [4]	DOPD [5]	PointFusion [6]	DenseFusion [7]	G2L-Net [8]	PVN3D [1]	FFB6D [9]	Our Uni6D
ape	77.0	43.6	64.4	87.7	70.4	92.3	96.8	97.3	98.4	93.71
benchvise	97.5	99.9	97.8	98.5	80.7	93.2	96.1	99.7	100.0	99.81
camera	93.5	86.9	91.7	96.1	60.8	94.4	98.2	99.6	99.9	95.98
can	96.5	95.5	95.9	99.7	61.1	93.1	98.0	99.5	99.8	99.02
cat	82.1	79.3	83.8	94.7	79.1	96.5	99.2	99.8	99.9	98.10
driller	95.0	96.4	96.2	98.8	47.3	87.0	99.8	99.3	100.0	99.11
duck	77.7	52.6	66.8	86.3	63.0	92.3	97.7	98.2	98.4	89.95
eggbox	97.1	99.2	99.7	99.9	99.9	99.8	100.0	99.8	100.0	100.00
glue	99.4	95.7	99.6	96.8	99.3	100.0	100.0	100.0	100.0	99.23
holepuncher	52.8	82.0	85.8	86.9	71.8	92.1	99.0	99.9	99.8	90.20
iron	98.3	98.9	97.9	100.0	83.2	97.0	99.3	99.7	99.9	99.49
lamp	97.5	99.3	97.9	96.8	62.3	95.3	99.5	99.8	99.9	99.42
phone	87.7	92.4	90.8	94.7	78.8	92.8	98.9	99.5	99.7	97.41
Avg	88.6	86.3	89.9	95.2	73.7	94.3	98.7	99.4	99.7	97.03

Table 3. Evaluation results (ADD-0.1d ACC) on the Occlusion-LineMOD dataset. Symmetric objects are denoted in bold.

Method	PoseCNN [2]	Oberweger [10]	Pix2Pose [11]	PVNet [3]	DPOD [5]	Hu [12]	HybridPose [13]	PVN3D [1]	FFB6D [9]	Our Uni6D
ape	9.6	12.1	22.0	15.8	-	19.2	20.9	33.9	47.2	32.99
can	45.2	39.9	44.7	63.3	-	65.1	75.3	88.6	85.2	51.04
cat	0.9	8.2	22.7	16.7	-	18.9	24.9	39.1	45.7	4.56
driller	41.4	45.2	44.7	65.7	-	69.0	70.2	78.4	81.4	58.40
duck	19.6	17.2	15.0	25.2	-	25.3	27.9	41.9	53.9	34.80
eggbox	22.0	22.1	25.2	50.2	-	52.0	52.4	80.9	70.2	1.73
glue	38.5	35.8	32.4	49.6	-	51.4	53.8	68.1	60.1	30.16
holepuncher	22.1	36.0	49.5	39.7	-	45.6	54.2	74.7	85.9	32.07
Avg	24.9	27.0	32.0	40.8	47.3	43.3	47.5	63.2	66.2	30.71

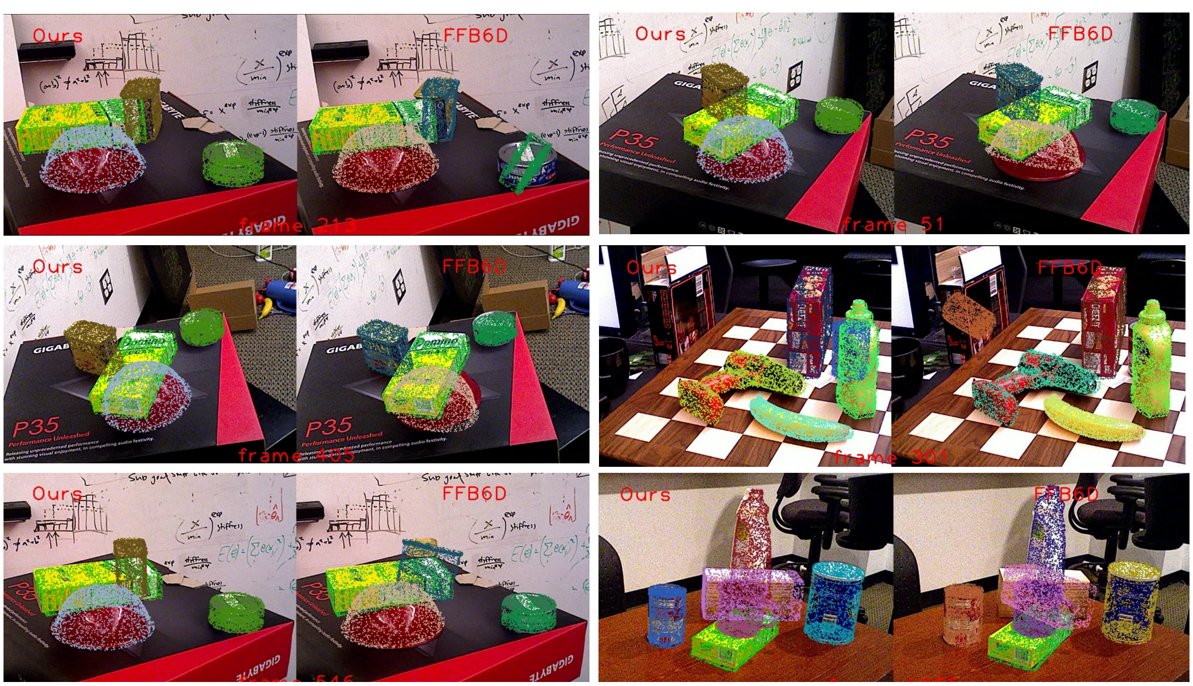

Figure 1. Qualitative results of 6D pose on the YCB-Video dataset. In each sub-figure, left is the result of our method and the right is of the SOTA method FFB6D [9].

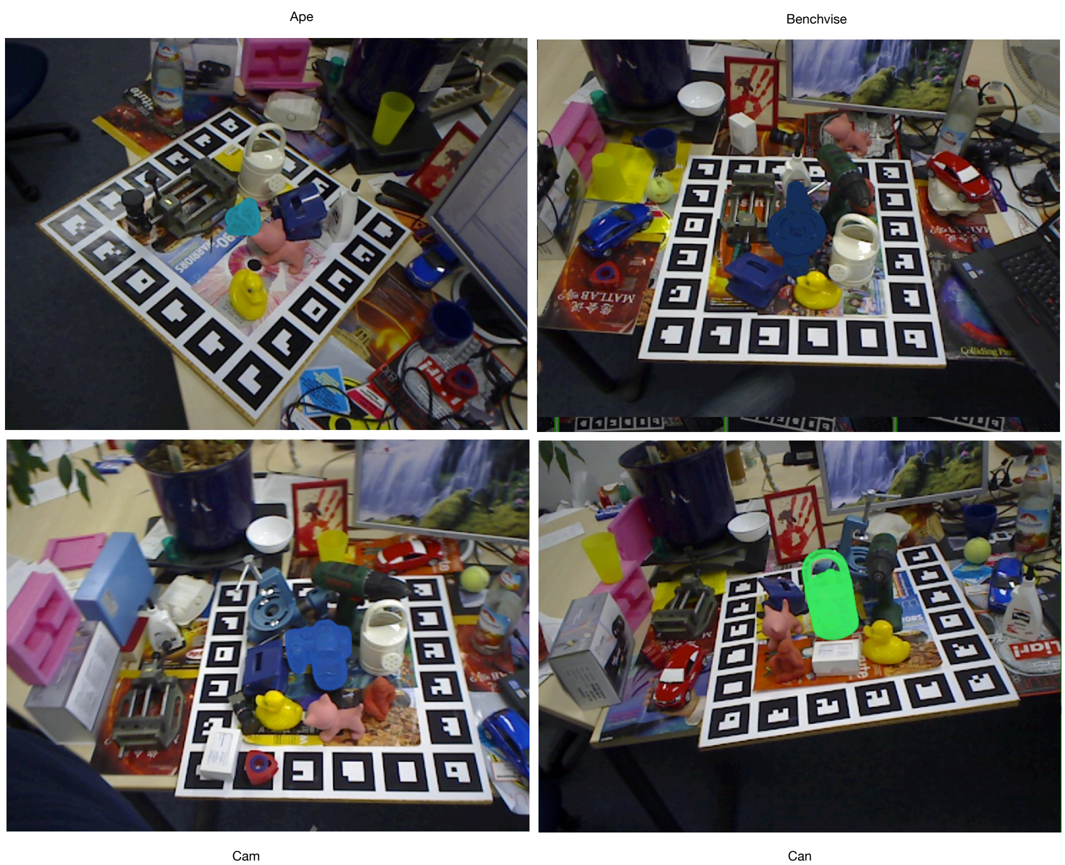

Figure 2. Qualitative results of 6D pose on the LineMOD dataset.