Supplementary Material - FFB6D: A Full Flow Bidirectional Fusion Network for 6D Pose Estimation

Yisheng He[1] Haibin Huang[3] Haoqiang Fan[2] Qifeng Chen[1] Jian Sun[2]
[1]Hong Kong University of Science and Technology [2] Megvii Technology [3]Kuaishou Technology

A. Details of the Network Architecture

Figure 1 shows the detailed architecture of the proposed FFB6D. We applied ImageNet [3] pre-trained ResNet34 [4] and PSPNet [21] as encoder and decoder of the input RGB image. Meanwhile, a RandLA-Net [7] is applied for point cloud representation learning. On each encoding and decoding layer of the two networks, point-to-pixel and pixel-to-point fusion modules are added for information communication. Finally, the extracted dense appearance and geometry features are concatenated and fed into the semantic segmentation, center point voting, and 3D keypoints voting modules for pose estimation. Details of each part are as follows:

Network Input. The input of the convolution neural network (CNN) branch is a full scene image with a size of $H \times W \times 3$, where H is the height of the RGB image, W the width, and 3 the three channels of color information (RGB). For the point cloud learning branch, the input is a randomly subsampled point cloud from the scene depth image, with a size of $N \times C_{in}$, where N set to 12288 is the number of sampled points, and C_{in} the input coordinate, color and normal information of each point (x-y-z-R-G-B-nx-ny-nz).

Encoding Layers. We utilize ResNet34 [4] as the encoder of RGB images, which consists of five convolution layers to reduce the size of feature maps and increase the number of feature channels. The Pyramid Pooling Modules (PPM) from PSPNet [21] is also applied in the last encoding layer. Meanwhile, in the point cloud network branch, after being processed by a fully connected layer, the point features are fed into four encoding layers of RandLA-Net [7] for feature encoding, each of which consists of a local feature aggregation module and a random sampling operation designed in the work [7].

Decoding Layers. In the decoding stage, three upsampling modules and a final convolution layer from PSPNet are used for appearance feature decoding. Meanwhile, in the point cloud network branch, four decoding layers from RandLA-Net are utilized as point cloud features decoders, which consists of the random sampling operations and local feature aggregation modules designed in [7].

Bidirectional Fusion Modules. On each encoding and decoding stage, point-to-pixel and pixel-to-point fusion modules (Section 3.2) are added for bidirectional information communication. For each pixel-to-point fusion module, we set $K_{r2p} = 16$ and aggregate 16 nearest pixel of appearance features through a max-pooling and a single layer shared MLP, $MLP[c_r, c_p]$, where c_r denotes the channel size of RGB features and c_p the channel size of corresponding point features. The aggregated pixels of appearance features are then concatenated with the corresponding point features and map by a shared MLP, $MLP[2 * c_p, c_p]$ to generate each fused point feature. Meanwhile, we set $K_{p2r} = 1$ and get the fused appearance features similarly in each point-to-pixel fusion module.

Prediction Headers. Three headers are added after the extracted dense RGBD features to predict the semantic label, center point offset as wel as the 3D keypoints offsets of each point. These headers consists of shared MLPs, denoted as $MLP[c_r{+}c_p, c_1, c_2, ..., c_k]$, where c_r and c_p represent the channel size of extracted appearance and geometry features respectively, and c_i the output channel size of the i-th layer in the MLP. Specifically, the semantic segmentation module consists of $MLP[c_r + c_p, 128, 128, 128, n_{cls}]$, the center offset learning module comprises $MLP[c_r + c_p, 128, 128, 128, 3]$, and the 3D keypoints offset module is composed of $MLP[c_r + c_p, 128, 128, 128, n_{kps} * 3]$, where n_{cls} denotes number of object classes and n_{kps} means the number of keypoints of each object.

Pose Estimation Modules. Given the predicted semantic label and center point offset of each point in the scene, a MeanShift [2] clustering algorithm is applied to distinguish different object instances with the same semantic. Then, for each instance, each point within it votes for its 3D keypoint with the MeanShift [2] algorithm. Finally, a least-squares fitting algorithm is applied to recover the object pose parameters according to the detected 3D keypoints.

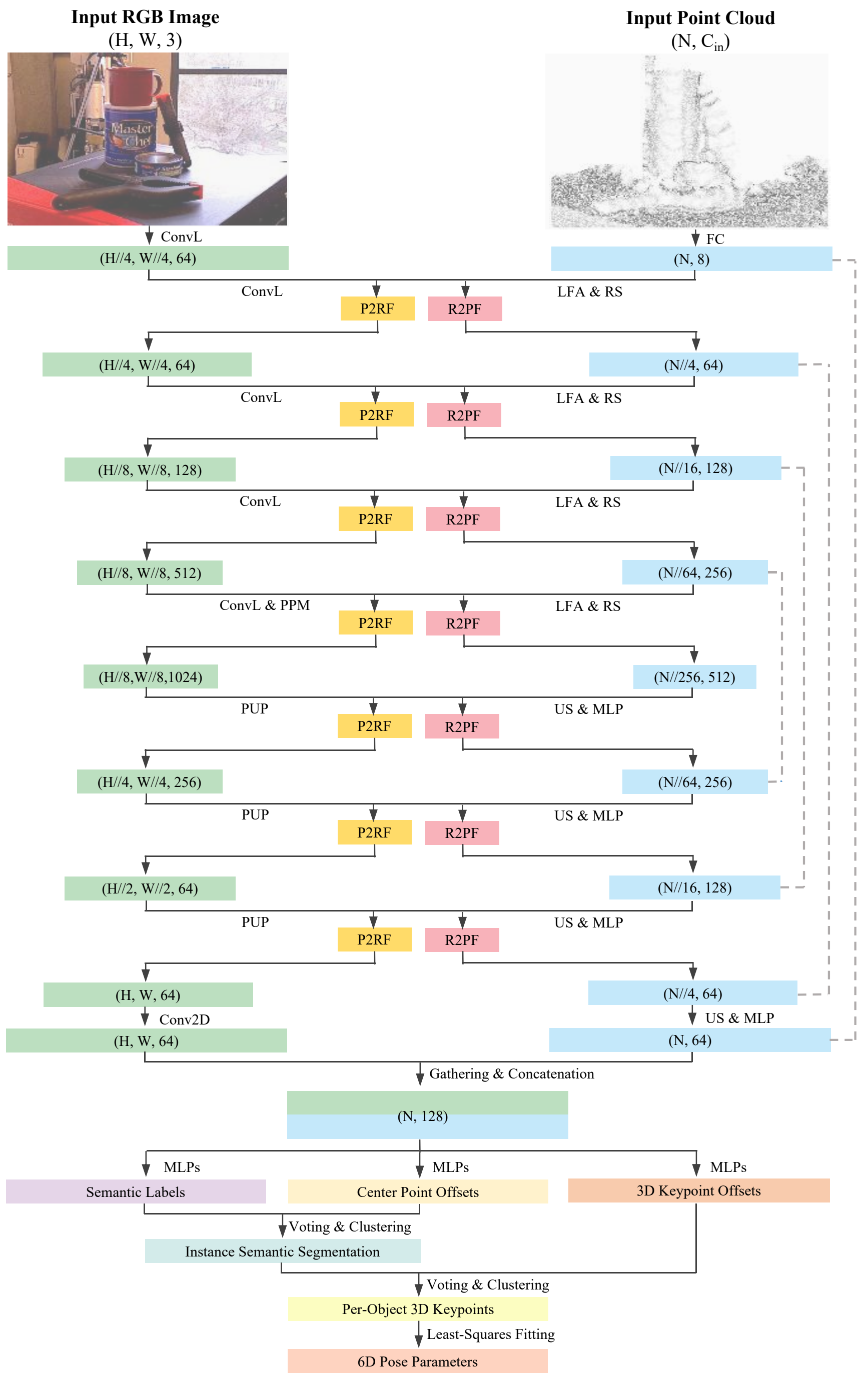

Figure 1: **The detailed architecture of our FFB6D.** For the convolution neural network (CNN) branch on the RGB image, we utilize ResNet34 [4] and PSPNet [21] as encoder and decoder. ConvL: Convolution Layers of ResNet34, PPM: Pyramid Pooling Modules of PSPNet, PUP: PSPNet Up-sampling, Conv2D: 2D convolution layer. For the point cloud network (PCN) branch on the point cloud, we apply RandLA-Net [7] for feature extraction. FC: Fully Connected layer, LFA: Local Feature Aggregation, RS: Random Sampling, MLP: shared Multi-Layer Perceptron, US: Up-sampling. In the flow of the two networks, point-to-pixel fusion modules, P2RF, and pixel-to-point fusion modules, R2PF consists of max pooling and shared MLPs are added. The extracted features from the two networks are then concatenated and fed into the following semantic segmentation, center point voting and 3D keypoints voting modules [5] composed by shared MLPs. A clustering algorithm is then applied to distinguish different instances with the same semantic labels and points on the same instance vote for their target keypoints. With detected 3D keypoints, a least-squares fitting algorithm is applied to recover the pose parameters.

	RGB				RGB-D				
	PoseCNN DeepIM [18, 10]	PVNet[15]	CDPN[11]	DPOD[20]	Point-Fusion[19]	Dense-Fusion[17]	G2L-Net[1]	PVN3D[5]	Our FFB6D
ape	77.0	43.6	64.4	87.7	70.4	92.3	96.8	97.3	**98.4**
benchvise	97.5	99.9	97.8	98.5	80.7	93.2	96.1	99.7	**100.0**
camera	93.5	86.9	91.7	96.1	60.8	94.4	98.2	99.6	**99.9**
can	96.5	95.5	95.9	99.7	61.1	93.1	98.0	99.5	**99.8**
cat	82.1	79.3	83.8	94.7	79.1	96.5	99.2	99.8	**99.9**
driller	95.0	96.4	96.2	98.8	47.3	87.0	99.8	99.3	**100.0**
duck	77.7	52.6	66.8	86.3	63.0	92.3	97.7	98.2	**98.4**
eggbox	97.1	99.2	99.7	99.9	99.9	99.8	**100.0**	99.8	**100.0**
glue	99.4	95.7	99.6	96.8	99.3	**100.0**	**100.0**	**100.0**	**100.0**
holepuncher	52.8	82.0	85.8	86.9	71.8	92.1	99.0	**99.9**	99.8
iron	98.3	98.9	97.9	100.0	83.2	97.0	99.3	99.7	**99.9**
lamp	97.5	99.3	97.9	96.8	62.3	95.3	99.5	99.8	**99.9**
phone	87.7	92.4	90.8	94.7	78.8	92.8	98.9	99.5	**99.7**
MEAN	88.6	86.3	89.9	95.2	73.7	94.3	98.7	99.4	**99.7**

Table 1: Quantitative evaluation on the LineMOD dataset. The ADD-0.1d [6] metric is reported and symmetric objects are in bold.

Method	PoseCNN [18]	Oberweger [13]	Hu et al. [9]	Pix2Pose [14]	PVNet [15]	DPOD [20]	Hu et al.[8]	HybridPose [16]	PVN3D [5]	Our FFB6D
ape	9.6	12.1	17.6	22.0	15.8	-	19.2	20.9	33.9	**47.2**
can	45.2	39.9	53.9	44.7	63.3	-	65.1	75.3	**88.6**	85.2
cat	0.9	8.2	3.3	22.7	16.7	-	18.9	24.9	39.1	**45.7**
driller	41.4	45.2	62.4	44.7	65.7	-	69.0	70.2	78.4	**81.4**
duck	19.6	17.2	19.2	15.0	25.2	-	25.3	27.9	41.9	**53.9**
eggbox	22.0	22.1	25.9	25.2	50.2	-	52.0	52.4	**80.9**	70.2
glue	38.5	35.8	39.6	32.4	49.6	-	51.4	53.8	**68.1**	60.1
holepuncher	22.1	36.0	21.3	49.5	39.7	-	45.6	54.2	74.7	**85.9**
MEAN	24.9	27.0	27.0	32.0	40.8	47.3	43.3	47.5	63.2	**66.2**

Table 2: Quantitative evaluation on the Occlusion-LineMOD dataset. The ADD-0.1d [6] metric is reported and symmetric objects are in bold.

B. Implementation: Different Representation Learning Frameworks

In this section, we demonstrate the implementation details of different representation learning frameworks in Table 4. To implement CNN-R$\oplus$D, we lift each pixel in the depth image to its corresponding 3D point to get the XYZ map as well as the normal map. We then concatenate them with the RGB map and feed it into a ResNet34-PSPNet encoding-decoding network for feature extraction of each point (pixel). For PCN-R$\oplus$D, we append RGB values of each point to its 3D coordinate as well as its normal vector and then utilize the RandLA-Net for representation learning. The CNN-R+CNN-D utilizes two ResNet34-PSPNet networks for feature extraction from the RGB image and the XYZ and normal maps respectively. Bidirectional fusion modules (Section 3.2) are added to each encoding and decoding layer. For PCN-R+PCN-D, we leverage one RandLA-Net to extract features from the RGB value of each point and another one for representation learning of the 3D coordinate and the normal vector of each point. In the network flow, bidirectional fusion modules are added to each layer for information communication as well. To implement CNN-R+3DC-D, we replace the RandLA-Net with a 3D convolution neural network. In the encoding stages, the voxel size decreases from 32^3 to 4^3 ($32^3 \rightarrow 16^3 \rightarrow 8^3 \rightarrow 4^3$) and the feature dimensions increases from 32 to 256 ($32 \rightarrow 64 \rightarrow 128 \rightarrow 256$). In the decoding stage, the voxel size increases and feature dimensions decrease inversely. Finally, the geometry feature of each point is obtained within a trilinear interpolation manner, as in PVCNN[12], which is then concatenated with the appearance feature from CNN.

C. More Results

C.1. Quantitative result on the LineMOD dataset.

More results of 6D pose estimation on the LineMOD dataset are shown in Table 1.

C.2. Quantitative result on the Occlusion-LineMOD dataset.

We report more results on the Occlusion-LineMOD dataset in Table 2. We follow the state-of-the-art to train our model on the LineMOD dataset and only use this dataset for testing.

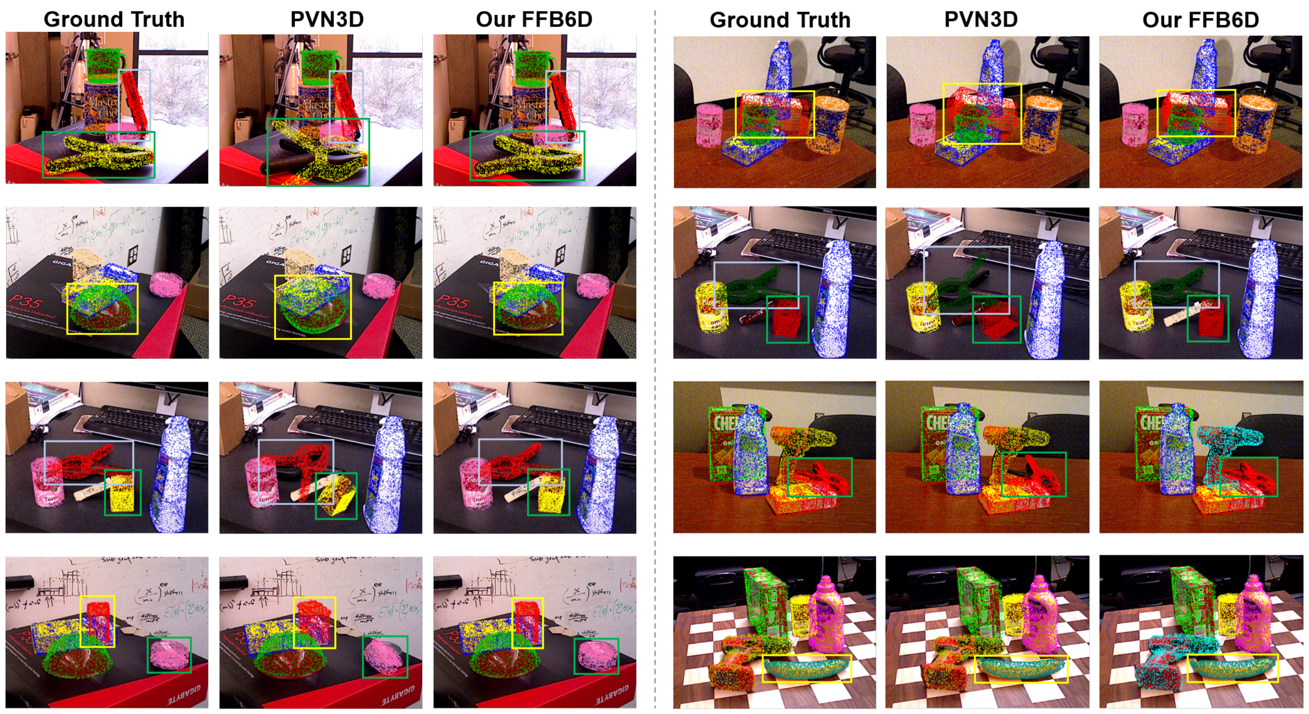

Figure 2: Qualitative results of 6D pose on the YCB-Video dataset. Objects in bounding boxes show the pose that we outperform the state-of-the-art significantly. Object vertexes in the object coordinate system are transformed by the ground truth or predicted pose to the camera coordinate system and then projected to the image by the camera intrinsic matrix. Compared to PVN3D [5] with the DenseFusion [17] architecture, our FFB6D is more robust towards occlusion and objects with similar appearance or reflective surfaces, which are quite challenging for either isolated CNN or point cloud network feature extraction.

C.3. Visualization on predicted pose on the YCB-Video Dataset.

We provide some qualitative results on the YCB-Video dataset in Figure 2.

References

[1] 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

[2] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5):603–619, 2002. 1

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 2

[5] 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. 2, 3, 4

[6] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer, 2012. 3

[7] 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, 2

[8] 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

[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. 3

[10] 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 (ECCV)*, pages 683–698, 2018. 3

[11] 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 International Conference on Computer Vision*, pages 7678–7687, 2019. 3

[12] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems*, pages 965–975, 2019. 3

[13] 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

[14] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. *arXiv preprint arXiv:1908.07433*, 2019. 3

[15] 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 Conference on Computer Vision and Pattern Recognition*, pages 4561–4570, 2019. 3

[16] 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

[17] 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 Conference on Computer Vision and Pattern Recognition*, pages 3343–3352, 2019. 3, 4

[18] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017. 3

[19] 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

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

[21] 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, 2