PR-GCN: A Deep Graph Convolutional Network with Point Refinement for 6D Pose Estimation

Guangyuan Zhou, Huiqun Wang, Jiaxin Chen, Di Huang

State Key Laboratory of Software Development Environment, Beihang University, Beijing, China
School of Computer Science and Engineering, Beihang University, Beijing, China
{zhouguangyuan, hqwangscse, jiaxinchen, dhuang}@buaa.edu.cn

Abstract

RGB-D based 6D pose estimation has recently achieved remarkable progress, but still suffers from two major limitations: (1) ineffective representation of depth data and (2) insufficient integration of different modalities. This paper proposes a novel deep learning approach, namely Graph Convolutional Network with Point Refinement (PR-GCN), to simultaneously address the issues above in a unified way. It first introduces the Point Refinement Network (PRN) to polish 3D point clouds, recovering missing parts with noise removed. Subsequently, the Multi-Modal Fusion Graph Convolutional Network (MMF-GCN) is presented to strengthen RGB-D combination, which captures geometry-aware intermodality correlation through local information propagation in the graph convolutional network. Extensive experiments are conducted on three widely used benchmarks, and state-of-the-art performance is reached. Besides, it is also shown that the proposed PRN and MMF-GCN modules are well generalized to other frameworks.

1. Introduction

6D pose estimation aims to predict the orientation and location of an object in the 3D space from a canonical frame. It has received extensive attention in computer vision, since it is one of the fundamental steps for a wide range of applications, such as robotics grasping and augmented reality. Traditional methods attempt to accomplish this task based on RGB images only. They adopt handcraft features (e.g., SIFT and SURF) to establish correspondence between input and canonical images. Inspired by the great success in detection/recognition, deep neural networks are recently explored to address this issue, including the single-stage regression methods and keypoint based methods. Despite the remarkable promotion in accuracy, RGB-based deep models heavily rely on textures; thus sensitive to illumination variations, severe occlusions, and cluttered backgrounds.

Along with the emergence and innovation of depth sensors, 6D pose estimation on RGB-D data has become popular, expecting to deliver performance gain by adding geometry information. Early works estimate object poses from RGB images and refine them according to depth maps. Later studies represent depth images as 3D point clouds and the models are more efficient in computation and storage than those on original depth maps. By jointly making use of both modalities, RGB-D based solutions report better scores, with the superiority in the presence of the difficulties aforementioned as well as the low-texture case.

However, current RGB-D pose estimation suffers from...
two major limitations: ineffective representation of depth data and insufficient combination of two modalities. For the former, as captured in cluttered scenes, depth information is usually noisy and incomplete (see Fig. 1). Inferring poses from such data, either in 2D depth maps or 3D point clouds, is not robust, leading to accuracy deterioration. For the latter, RGB and depth clues are fused by concatenating separately learned single-modal features [38] or by applying a simple point-wise encoder [9], where inter-modality correlations are not considered or roughly modeled in a global manner, leaving much room for improvement.

In this paper, we propose a novel deep learning approach, namely Graph Convolutional Network with Point Refinement (PR-GCN), to simultaneously address the two limitations in a unified way. As in Fig. 2, given the RGB image and 3D point cloud (generated from depth map) of an object, we first introduce a Point Refinement Network (PRN) to polish the point cloud. Endowed with an encoder-decoder structure and trained with a regularized multi-resolution regression loss, PRN recovers the missing parts of the raw input with noise removed. Subsequently, we integrate RGB-D clues by a Mutli-Modal Fusion Graph Convolutional Network (MMF-GCN). It constructs a \( k \)-Nearest Neighbor (\( k \)-NN) graph and extracts geometry-aware inter-modality correlation through local information propagation in the Graph Convolutional Network (GCN). An additional \( k \)-NN graph and GCN are employed to encode local geometry attributes of the refined point cloud as a complement to the original data. The features from the two GCNs are then combined and fed into several fully-connected layers for final 6D pose prediction. We extensively evaluate PR-GCN on three public benchmarks, Linemod [11], Occlusion Linemod [2], and YCB-Video [40], and achieve the state-of-the-art performance. We also show that the proposed PRN and MMF-GCN modules are well generalized to other frameworks.

The contributions: 1) We propose the PR-GCN approach to 6D pose estimation by enhancing depth representation and multi-modal combination. 2) We present the PRN module with a regularized multi-resolution regression loss for point-cloud refinement. To the best of our knowledge, it is the first that applies 3D point generation to this task. 3) We develop the MMF-GCN module to capture local geometry-aware inter-modality correlation for RGB-D fusion.

2. Related Work


RGB-D based 6D Pose Estimation. With geometry information, depth maps contribute to pose estimation for various lighting conditions and low-textured appearances, complementary to RGB images. MCN [17] employs two CNNs for representation learning in RGB and depth respectively and resulting features are then concatenated for pose prediction. PoseCNN [40] and SSD-6D [14] follow the coarse-to-fine scheme, where poses are initially estimated on RGB frames and subsequently refined on depth maps. [37] builds a multi-view model to jointly reconstruct whole scenes and optimize multi-object poses.

Recently, there has emerged a trend to represent geometry clues in 3D point clouds rather than depth maps for higher efficiency [38, 5, 4, 31, 9]. DenseFusion [38] designs a heterogeneous network to integrate texture and shape features and such representation proves more discriminative than single-modal ones. CF [5] introduces attention modules to combine the two modalities for further improvements. G2L [4] segments point clouds of objects in scenes by frustum pointnet [31] and regresses pose parameters via extra coordinate constraints. PVN3D [9] incorporates DenseFusion into 3D key-point detection and instance semantic segmentation, significantly boosting the performance. Unfortunately, the point clouds generated from the depth maps are often of a low quality, since the shape information is often incomplete and noisy as Fig. 1 shows. Besides, the combination of RGB and depth clues is launched in a very rough way, e.g., directly concatenating or pointwise encoding. In contrast, our approach develops the PRN and MMF-GCN modules to polish depth clues by generating refined point clouds and enhance integration by capturing local geometry-aware inter-modality correlations respectively, both of which are beneficial to pose estimation.

3. The Proposed Method

3.1. Framework Overview

RGB-D based 6D pose estimation recovers 6D poses of objects in RGB-D images, where 6D pose is usually represented by a rotation matrix \( R \in SO(3) \) and a translation vector \( t \in \mathbb{R}^3 \). For this issue, we propose the PR-GCN approach. As Fig. 2 depicts, it consists of four steps: object localization and 3D points generation, 3D points refinement, GCN-based multi-modal fusion, and 6D pose prediction.
Object Localization and 3D Points Generation. Given an RGB-D image \( I = (I_{rgb}, I_d) \), we firstly locate objects on \( I_{rgb} \) using the off-the-shelf Faster R-CNN [33] detector, where \( I_{rgb} \) and \( I_d \) denote the RGB and depth channels of \( I \). According to the detected bounding boxes, we crop the sub-images \( \{ I_o \} = \{ (I_{o,rgb}, I_{o,d}) \} \), each of which contains an instance \( o \). \( I_{o,rgb} \) and \( I_{o,d} \) are the RGB and depth channels of \( I_o \). As in PoseCNN [40], we add a segmentation head to remove background of \( I_{o,rgb} \). With \( M_o \) (foreground mask) and \( I_{o,d} \), raw 3D points \( P_o = \{ p_o^{(1)}; \ldots ; p_o^{(i)}; \ldots ; p_o^{(N)} \} \in \mathbb{R}^{N \times 3} \) are rendered by Point-Cloud Transform (PCT) [8], where \( N \) is the number of points and \( p_o^{(i)} \in \mathbb{R}^3 \) is the 3D coordinate of the \( i \)-th point. It is worth noting that \( P_o \) could be severely incomplete and noisy due to external occlusions and sensor noise (see Fig. 1).

3D Points Refinement. To polish the quality of the generated raw 3D points \( P_o \), we propose the PRN module. As in Fig. 2, it is composed of an MLP-based encoder and a multi-resolution decoder to recover the complete and accurate 3D point cloud \( \hat{P}_o = \{ \hat{p}_o^{(1)}; \ldots ; \hat{p}_o^{(m)}; \ldots ; \hat{p}_o^{(M)} \} \in \mathbb{R}^{M \times 3} \), where \( M \) is the number of refined points. A regularized multi-resolution regression loss is formulated in training, enhancing its ability of filtering out noise in \( P_o \).

GCN-based Multi-Modal Fusion. For more sufficient RGB-D fusion, we propose the MMF-GCN module. As in Fig. 2, it extracts texture and geometry features from \( I_{o,rgb} \) and \( P_o \), respectively, and a graph is built based on geometry distribution. Accordingly, \( I_{o,rgb} \) and \( P_o \) are initially integrated by applying a GCN \( GCN_f(\cdot) \) on the previously built graph through local information propagation. The geometry clues from the refined 3D points are encoded by introducing an extra GCN \( GCN_{ref}(\cdot) \) and then incorporated into the initially fused features, which are fed into several stacked fully-connected layers \( T(\cdot) \) for further fusion. The resulting feature \( G_o = [g_o^{(k)}]_{k=1,\ldots,K} \in \mathbb{R}^{K \times d} \) is therefore the multi-modal representation for successive 6D pose estimation, where \( K \) and \( d \) refer to the number of points and the feature dimension, respectively. Since MMF-GCN captures local geometry-aware inter-modality correlation and leverages refined 3D point clouds, it is expected to deliver more discriminative and robust features.

6D Pose Prediction. \( [g_o^{(k)}]_{k=1,\ldots,K} \) is finally fed into three regression branches: \( REG_r(\cdot), REG_t(\cdot), REG_e(\cdot) \), for rotations \( \{ \hat{R}_o^{(k)} = REG_r(g_o^{(k)}) \} \), translations \( \{ \hat{t}_o^{(k)} = REG_t(g_o^{(k)}) \} \), confidence scores \( \{ s_o^{(k)} = REG_e(g_o^{(k)}) \} \), respectively. Each branch has four fully-connected layers. Similar to [9, 29, 38], we select the candidate with the highest confidence score as the estimated pose, formulated as:

\[
(\hat{R}_o, \hat{t}_o) = \arg\max \{ s_o^{(k)} | k=1,\ldots,K \}
\] (1)

3.2. Point Refinement Network

Recall that PRN aims to generate the refined 3D point cloud \( \hat{P}_o \) from the raw one of a low quality \( P_o \). As in Fig. 3, PRN is endowed with an encoder-decoder architecture. To deal with the change in point density (resolution), we downsample \( P_o \) with 1/2 and 1/4 scales, resulting in two extra point clouds: \( P_{o,1/2} \in \mathbb{R}^{2 \times 3} \) and \( P_{o,1/4} \in \mathbb{R}^{4 \times 3} \).

Accordingly, the encoder \( E(\cdot) \) has three branches \( E_1(\cdot), E_{1/2}(\cdot), E_{1/4}(\cdot) \), whose inputs are \( P_o, P_{o,1/2}, P_{o,1/4} \) and outputs are the three representations \( v_1 = E_1(P_o), v_{1/2} = E_{1/2}(P_o), v_{1/4} = E_{1/4}(P_o) \).
$E_{1/2}(P_{o,1/2}), v_{1/4} = E_{1/4}(P_{o,1/4})$. Each branch is a stack of six MLP layers. The concatenation of $v_1, v_{1/2}$ and $v_{1/4}$ followed by an MLP layer forms the intermediate latent representation $v$.

The decoder $DEC(·)$ employs a multi-resolution structure as [43] does. The first branch with four fully-connected (FC) layers as well as one reshape (RS) operation is built to obtain the coarse point cloud of a low-resolution $P_{o,1/8} = RS_8(FC_4(FC_3(FC_2(FC_1(v))))) \in \mathbb{R}^{4 \times 3}$ (or $\mathbb{R}^{8 \times 3 \times 1}$ equivalently). The second branch consists of the first two shared FC layers, one additional FC layer, one MLP layer and one RS operation, generating a medium-resolution point cloud $P_{o,1/4} = RS_4(MLP_{2,2}(FC_{2,1}(FC_2(FC_1(v)))))$. Then, $P_{0,1/8}$ is integrated into $P_{o,1/4}$ by a broadcasting additive operation $P_{o,1/4} := P_{o,1/4} \oplus P_{0,1/8}$, and $P_{o,1/4}$ is reshaped into $P_{0,1/4} := RS_{2,4}(P_{o,1/4}) \in \mathbb{R}^{4 \times 3}$. Similarly, the third branch finally renders a high-resolution point cloud $P_o \in \mathbb{R}^{M \times 3}$ by sharing the first FC layer, adding one FC layer, three MLP layers and one RS operation, and incorporating the multi-resolution information as $P_o = RS_{1,7}(RS_{1,6}(MLP_{1,5}(MLP_{1,4}(MLP_{1,3}(FC_{1,2}(FC_1(v))))))) \oplus P_{0,1/4}$.

For training PRN, we develop a multi-resolution regression loss formulated as follows:

$$L_{mr} = d_C(P_o, P_{o,GT}) + \sum_{r \in \{1/8\}} d_C(P_{o,r}, P_{o,GT}) \tag{2}$$

where $P_{o,GT} = \{P_{o,GT}^{(1)}, \ldots, P_{o,GT}^{(H)}\} \in \mathbb{R}^{H \times 3}$ is ground-truth point cloud of object $o$. $d_C(·, ·)$ is the Chamfer distance defined as $d_C(P, Q) = \frac{1}{M} \sum_j \min_i ||p^{(i)} - q^{(j)}||_2^2 + \frac{1}{N} \sum_i \min_j ||q^{(j)} - p^{(i)}||_2^2$, given $P = [p^{(m)}]_{m=1, \ldots, M} \in \mathbb{R}^{M \times 3}$ and $Q = [q^{(n)}]_{n=1, \ldots, N} \in \mathbb{R}^{N \times 3}$.

In Eq. (2), $P_o$ is forced to fit the ground-truth in low- to high-resolutions when minimizing $L_{mr}$. In other words, $DEC(·)$ is forced to predict high-quality points in multiple resolutions by a unified structure, and thus optimized with more supervision than the single-resolution case. Moreover, as shown in Fig. 3, the high-resolution output $P_o$ integrates multi-resolution information from $\hat{P}_{o,1/4}$ and $\hat{P}_{o,1/8}$. As a consequence, PRN mitigates the incompleteness and decreases the noise of the raw 3D points.

Despite the aforementioned advantages of $L_{mr}$, it fails to perceive the global point distribution of $P_{o,GT}$. We handle this problem by introducing the adversarial loss:

$$L_{adv} = \sum_{h=1}^{H} \log(D(p^{(h)}_{o,GT})) + \sum_{m=1}^{M} \log(1 - D(p^{(m)}_o)), \tag{3}$$

where $D(·)$ is the discriminator to classify whether a point belongs to $P_{o,GT}$ (“real”) or not (“fake”). By minimizing $L_{adv}$, PRN is expected to generate $P_o$ that captures holistic point distribution of $P_{o,GT}$, benefiting the quality of $P_o$.

The regularized multi-resolution regression loss is thus formulated as:

$$L_{prn} = \sum_o (\lambda \cdot L_{adv} + \beta \cdot L_{mr}), \tag{4}$$

where $\lambda$ and $\beta$ are the trade-off hyper-parameters.

### 3.3. Multi-Modal Fusion GCN

As mentioned before, given the RGB ($I_{o,rgb}$) and point cloud ($P_o$ and $P_o$) data of object $o$, MMF-GCN integrates multi-modal information into more effective representation ($G_o$) for accurate 6D pose estimation.

Specifically, MMF-GCN first extracts the geometry feature $f^{(i)}_{o,rgb}$ from $P_o$ and the texture feature $f^{(i)}_{o,rgb}$ from $I_{o,rgb}$ for the $i$-th point $p^{(i)}_o \in P_o$. The normalized coordinate of $p^{(i)}_o$ is directly used as the geometry feature, and by mapping this coordinate to the pixel on $I_{o,rgb}$, PSPNet [46] with the ResNet-18 backbone is adopted to compute pixel-wise representation as the texture feature.

When $\{f^{(i)}_{o,rgb}\}$ and $\{f^{(i)}_{o,rgb}\}$ are ready, a $k$-Nearest Neighbour ($k$-NN) graph $G_f = (V_f, E_f)$ is constructed. $V_f = \{p^{(i)}_o, \ldots, p^{(N)}_o\}$ and $E_f = \{(p^{(i)}_o, p^{(j)}_o) \mid p^{(i)}_o \in N_k(p^{(j)}_o)\}$ denote the vertices and the edges, and $N_k(p^{(i)}_o)$ indicates the $k$ nearest neighbors of $p^{(i)}_o$. The edge feature is defined as $e^{(i,j)} = h_{\theta}(f^{(i)}_o - f^{(j)}_o, f^{(i)}_o)$ with $f^{(i)}_o = [f^{(i)}_{o,rgb}, f^{(i)}_{o,rgb}]$, where $h_{\theta}(·, ·)$ is a nonlinear function parameterized by $\theta$.

Afterwards, a graph convolution network $GCN_f(·)$ is employed to capture local inter-modality correlations, with EdgeConv [39] for graph convolutions. The basic updating scheme is formulated as:

$$g^{(i)}_{o,f} = MP \left( h_{\theta(i-1)} \left( g^{(i-1)}_{o,f} - g^{(j-1)}_{o,f} \cdot g^{(i-1)}_{o,f} \right) \right),$$

where $g^{(i)}_{o,f}$ denotes the $i$-th edge feature in the $l$-th layer, $h_{\theta(i-1)}(·, ·)$ is a nonlinear function in the $(l-1)$-th layer, and $MP(·)$ refers to max pooling. The representation $G_{o,f} = [g^{(j)}_{o,f}]_{j=1, \ldots, J} \in \mathbb{R}^{J \times d_f}$ is then obtained, where $G_{o,f} =$...
Table 1. Comparison with the state-of-the-arts in terms of ADD(-S) (%) on Linemod. Symmetric objects are marked in bold. */† indicates that the method only uses real/synthetic data for training. PF and DF refer to Point Fusion [41] and DenseFusion [38], respectively.

<table>
<thead>
<tr>
<th>Object</th>
<th>PoseCNN*</th>
<th>PVNet</th>
<th>CDPN</th>
<th>DPPD</th>
<th>DPVL</th>
<th>PF*</th>
<th>SSD6D†</th>
<th>DF*</th>
<th>PVPN3D</th>
<th>G2L†</th>
<th>Ours</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>ape</td>
<td>77.0</td>
<td>43.6</td>
<td>64.4</td>
<td>87.7</td>
<td>69.1</td>
<td>70.4</td>
<td>65.0</td>
<td>92.3</td>
<td>97.3</td>
<td>96.8</td>
<td>97.6</td>
<td>99.2</td>
</tr>
<tr>
<td>benchvise</td>
<td>97.5</td>
<td>99.9</td>
<td>97.8</td>
<td>98.5</td>
<td>100.0</td>
<td>80.7</td>
<td>80.0</td>
<td>93.2</td>
<td>99.7</td>
<td>96.1</td>
<td>99.2</td>
<td>99.8</td>
</tr>
<tr>
<td>camera</td>
<td>93.5</td>
<td>86.9</td>
<td>91.7</td>
<td>96.1</td>
<td>94.1</td>
<td>60.8</td>
<td>78.0</td>
<td>94.4</td>
<td>99.6</td>
<td>98.2</td>
<td>99.4</td>
<td>100.0</td>
</tr>
<tr>
<td>can</td>
<td>96.5</td>
<td>95.5</td>
<td>95.9</td>
<td>99.7</td>
<td>98.5</td>
<td>61.1</td>
<td>86.0</td>
<td>93.1</td>
<td>99.5</td>
<td>98.0</td>
<td>98.4</td>
<td>99.4</td>
</tr>
<tr>
<td>cat</td>
<td>82.1</td>
<td>79.3</td>
<td>83.8</td>
<td>94.7</td>
<td>83.1</td>
<td>79.1</td>
<td>70.0</td>
<td>96.5</td>
<td>99.8</td>
<td>99.2</td>
<td>98.7</td>
<td>99.8</td>
</tr>
<tr>
<td>driller</td>
<td>95.0</td>
<td>96.4</td>
<td>96.2</td>
<td>98.8</td>
<td>99.0</td>
<td>47.3</td>
<td>73.0</td>
<td>87.0</td>
<td>99.3</td>
<td>99.8</td>
<td>98.8</td>
<td>99.8</td>
</tr>
<tr>
<td>duck</td>
<td>77.7</td>
<td>52.6</td>
<td>66.8</td>
<td>86.3</td>
<td>63.5</td>
<td>63.0</td>
<td>66.0</td>
<td>92.3</td>
<td>98.2</td>
<td>97.7</td>
<td>98.9</td>
<td>98.7</td>
</tr>
<tr>
<td>eggbox</td>
<td>97.1</td>
<td>99.2</td>
<td>99.7</td>
<td>99.9</td>
<td>100.0</td>
<td>99.9</td>
<td>100.0</td>
<td>99.8</td>
<td>100.0</td>
<td>99.9</td>
<td>99.9</td>
<td>99.6</td>
</tr>
<tr>
<td>glue</td>
<td>99.4</td>
<td>95.7</td>
<td>99.6</td>
<td>96.8</td>
<td>98.0</td>
<td>99.3</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>holepuncher</td>
<td>52.8</td>
<td>82.0</td>
<td>85.8</td>
<td>86.9</td>
<td>88.2</td>
<td>71.8</td>
<td>49.0</td>
<td>92.1</td>
<td>99.9</td>
<td>99.0</td>
<td>99.4</td>
<td>99.8</td>
</tr>
<tr>
<td>iron</td>
<td>98.3</td>
<td>98.9</td>
<td>97.9</td>
<td>100.0</td>
<td>99.9</td>
<td>83.2</td>
<td>78.0</td>
<td>97.0</td>
<td>99.7</td>
<td>99.3</td>
<td>98.5</td>
<td>99.5</td>
</tr>
<tr>
<td>lamp</td>
<td>97.5</td>
<td>99.3</td>
<td>97.9</td>
<td>96.8</td>
<td>99.8</td>
<td>62.3</td>
<td>73.0</td>
<td>95.3</td>
<td>99.8</td>
<td>99.5</td>
<td>99.2</td>
<td>100.0</td>
</tr>
<tr>
<td>phone</td>
<td>87.7</td>
<td>92.4</td>
<td>90.8</td>
<td>94.7</td>
<td>96.4</td>
<td>78.8</td>
<td>79.0</td>
<td>92.8</td>
<td>99.5</td>
<td>98.9</td>
<td>98.4</td>
<td>99.7</td>
</tr>
<tr>
<td>MEAN</td>
<td>88.6</td>
<td>86.3</td>
<td>89.9</td>
<td>95.2</td>
<td>91.5</td>
<td>73.7</td>
<td>79.0</td>
<td>94.3</td>
<td>99.4</td>
<td>98.7</td>
<td>98.9</td>
<td>99.6</td>
</tr>
</tbody>
</table>

GCN_f(\{f_{o,rgb}^{(i)}, f_{o,d}^{(i)}\}); J and d_f are the point number and the feature dimension, respectively.

As P_o is usually incomplete and noisy, MMF-GCN encodes the geometry attribute of P_o and incorporates it into G_o,f as a complement. Concretely, similar to G_f, another k-NN graph G_{o,ref} is built based on P_o and an extra GCN G_{o,ref}(·) is employed. The refined geometry feature is calculated using EdgeConv: \[ G_{o,ref} = [g_{o,ref}^{(j)}]_{k=1,...,J} \in \mathbb{R}^{J \times d_{ref}}, \] where G_{o,ref} = G_{o,ref}(P_o) and d_{ref} is the feature dimension. G_{o,ref} is subsequently combined with G_o through simple concatenation, which is further integrated by a few stacked FC layers T(·). At last, multi-modal representation is formed as \( G_o = T([G_{o,r}, G_{o,ref}]) \) for 6D pose estimation.

### 3.4. Training Objectives

The objective function for training PR-GCN consists of two parts: the pose estimation loss L_pose and the regularized multi-resolution regression loss L_prn, as depicted in Eq. (4).

Given the ground-truth 6D pose (\( R_o, t_o \)) and the predictions \( \{(\hat{R}_{o}^{(k)}, \hat{t}_{o}^{(k)}, s_{o}^{(k)})\} \) at K points \( \{x_{o}^{(k)}\} \), the pose estimation error of the i-th prediction \( (\hat{R}_{o}^{(i)}, \hat{t}_{o}^{(i)}) \) is defined as \( e_{o}^{(i)} = \frac{1}{K} \sum_{j=1}^{K} \min_{k} \| (R_{o}x_{o}^{(j)} + t_{o}) - (\hat{R}_{o}^{(i)}x_{o}^{(k)} + \hat{t}_{o}^{(i)}) \|^2 \).

Based on \( e_{o}^{(i)} \), we adopt an extra regularization term on the prediction scores \( \{s_{o}^{(i)}\} \) as in [38] and formulate the pose estimation loss as:

\[ L_{pose} = \frac{1}{K} \sum_{o} \sum_{i} e_{o}^{(i)} \cdot \left( s_{o}^{(i)} - \log(s_{o}^{(i)}) \right). \]

By combining Eq. (5) and Eq. (4), the overall training objective function is written as:

\[ \mathcal{L} = L_{pose} + \mu \cdot L_{prn}, \]

where \( \mu \) is the trade-off hyper-parameter.

### 4. Experiments

#### 4.1. Datasets and Metrics

Extensive evaluation is made on three datasets: Linemod [11], Occlusion Linemod [2] and YCB-Video [40].

**Linemod** [11] is composed of 15 RGB-D videos of 15 low-textured objects. Following [32], 13 objects are considered and the standard training/testing split is adopted as in [38, 40]. **Occlusion Linemod** is collected by annotating a subset of Linemod (8 out of 15 objects), where each image has multiple occluded objects, making it more challenging. **YCB-Video** [40] includes 21 objects with various textures and sizes. It provides RGB-D images and detailed pose annotations. There are 130K real images from 92 videos and 80K synthetically rendered ones, and 16,189 real and all the synthesized images are used in training, according to [38].

As in the literature, two main metrics are employed for evaluation, i.e., Average Distance (ADD) [40] and ADD-Symmetric (ADD-S) [40], designed for general objects and symmetric objects, respectively. DenseFusion [38] gives the ADD-S smaller than 2 centimeters (ADD-S<2cm) for real applications e.g., robotic manipulation, and PoseCNN [40] and DenseFusion [38] report the Area Under the ADD-S Curve (AUC) with the maximum threshold at 0.1m. We also show them for comparison.
4.2. Implementation Details

We fix the size of RGB images as 480×640. The numbers of raw/refined 3D points, i.e., \( N/M \), are set to 100/512 and 100/1,024 on Linemod (Occlusion Linemod) and YCB-Video, respectively. In MMF-GCN, refined point clouds are down-sampled to 100 points by FPS. When building the graphs \( \mathcal{G}_f \) and \( \mathcal{G}_{ref} \), we utilize 30 nearest neighbors, i.e., \( k = 30 \). In PRN, the hyper-parameters \( \lambda, \beta \) and \( \mu \) in the overall training loss \( \mathcal{L} \) are set to 0.05, 0.95 and 1.0. To train PR-GCN in a more stable way, PRN and MMF-GCN are progressively optimized. For instance, on YCB-Video, PRN and MMF-GCN are first alternatively trained for 15 epochs and then jointly optimized for 30 epochs.

In PRN training, we adopt the ADAM optimizer with the learning rate of 0.0001 and the batch size of 48. The rest parts of PR-GCN are trained for 40, 20 and 60 epochs on Linemod, Occlusion Linemod and YCB-Video, respectively, where the learning rate is initially set to 0.0001 and decayed by a factor of 0.3 after half of the maximal epochs.

4.3. Comparison with the State-of-the-art Methods

Results on Linemod. We first compare PR-GCN to the state-of-the-art methods on Linemod, including the RGB-based models: PoseCNN (+DeepIM) [40, 18], PVNet [29], CDPN [19], DPOD [44] and DPVL [42] and the RGB-D based ones: Point Fusion [41], SSD6D (+ICP) [14], Dense Fusion [38], PVN3D [9] and G2L [4]. Several approaches,
Table 4. Inference time of Segmentation (Seg), Point Refinement (PR), Pose Estimation (PE) and full PR-GCN (Full) on Linemod.

<table>
<thead>
<tr>
<th>Component</th>
<th>Seg (s)</th>
<th>PR (s)</th>
<th>PE (s)</th>
<th>Full (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>0.030</td>
<td>0.008</td>
<td>0.030</td>
<td>0.068</td>
</tr>
</tbody>
</table>

Table 5. Ablation study of PR-GCN in ADD-(S) (%) on Linemod.

<table>
<thead>
<tr>
<th>Method</th>
<th>PRN</th>
<th>MMF-GCN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline (with DGCNN)</td>
<td>×</td>
<td>×</td>
<td>94.8</td>
</tr>
<tr>
<td>Baseline+PRN</td>
<td>√</td>
<td>×</td>
<td>96.8</td>
</tr>
<tr>
<td>Baseline+MMF-GCN</td>
<td>×</td>
<td>√</td>
<td>96.9</td>
</tr>
<tr>
<td>Full model</td>
<td>√</td>
<td>√</td>
<td>98.9</td>
</tr>
</tbody>
</table>

Table 6. Generalization of PRN and MMF-GCN to other frameworks in terms of ADD-S (%) and <2cm (%) on YCB-Video.

<table>
<thead>
<tr>
<th>Method</th>
<th>PVN3D</th>
<th>DenseFusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td>ADD-S</td>
<td>&lt;2cm</td>
</tr>
<tr>
<td>Original model</td>
<td>95.5</td>
<td>97.6</td>
</tr>
<tr>
<td>w/ PRN</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>w/ MMF-GCN</td>
<td>96.2</td>
<td>98.4</td>
</tr>
<tr>
<td>w/ both</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Results on Occlusion Linemod. To evaluate the robustness of PR-GCN to inter-object occlusions, we display detailed results on Occlusion Linemod, in comparison with PoseCNN [40], DeepHeat [26], SS [12], Pix2Pose [28], PVNet [29], HybridPose [34] and PVN3D [9]. As shown in Table 2, our method consistently reaches the top ADD-(S) AUC and achieves the best mean ADD-(S) AUC, highlighting its superiority in the presence of heavy occlusions.

Results on YCB-Video. We then extend our analysis on this database and compare PR-GCN with PoseCNN (+ICP) [40], DenseFusion [38], PVN3D [9], CF [5] and G2L [4]. Table 3 shows the AUC and ADD-S<2cm for various methods. It can be observed that our method achieves the highest performance on both the metrics. For instance, compared to PVN3D and DenseFusion, PR-GCN improves the ADD-S<2cm by 0.9% and 1.7%, respectively.

Qualitative results. We additionally provide qualitative results in Fig. 4(a), comparing to DenseFusion and PVN3D. Due to cluttered backgrounds and severe occlusions, DenseFusion and PVN3D predict inaccurate poses in many cases, while our PR-GCN performs more robustly with much better results. We also demonstrate failure cases in Fig. 4(b), revealing that PR-GCN fails when dealing with extremely occluded objects and some symmetric ones.

Inference efficiency. Besides the accuracy, we evaluate the efficiency of our method on Linemod. As shown in Table 4, each key component infers fast, and the full pipeline takes 68ms on an Nvidia 1080Ti GPU, which is acceptable in downstream tasks such as robotic grasping.

4.4. Ablation Study

We comprehensively validate individual components of PR-GCN in the following.
Table 7. The influence of segmentation on different frameworks on YCB-Video in terms of AUC (%) and <2cm (%) (‘-’ indicates that the result is not reported).

<table>
<thead>
<tr>
<th></th>
<th>PoseCNN segmentation</th>
<th>PVN3D segmentation</th>
<th>GT segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PoseCNN</td>
<td>DenseFusion</td>
<td>Ours</td>
</tr>
<tr>
<td>AUC</td>
<td>93.0</td>
<td>93.1</td>
<td><strong>95.0</strong></td>
</tr>
<tr>
<td>&lt;2cm</td>
<td>93.2</td>
<td>96.8</td>
<td><strong>97.6</strong></td>
</tr>
</tbody>
</table>

Table 8. Ablation study of the multi-resolution loss on YCB-Video in terms of ADD-S (%) and <2cm (%).

<table>
<thead>
<tr>
<th></th>
<th>WO-PRN</th>
<th>PRN-SR</th>
<th>PRN-MR</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD-S</td>
<td>94.0</td>
<td>94.6</td>
<td><strong>95.8</strong></td>
</tr>
<tr>
<td>&lt;2cm</td>
<td>97.1</td>
<td>96.6</td>
<td><strong>98.5</strong></td>
</tr>
</tbody>
</table>

The impact of PRN and MMF-GCN. The baseline method removes PRN and replaces MMF-GCN by DGCNN [39] which adopts the same basic point cloud aggregator as our PR-GCN. As Table 5 displays, PRN boosts the baseline by 2.0%, indicating that refined point clouds contribute to pose estimation, while MMF-GCN achieves an improvement of 2.1%, demonstrating its advantage in integrating multi-modal features. The combination of PRN and MMF-GCN further enhances the performance.

The generalizability of PRN and MMF-GCN. We generalize the PRN and MMF-GCN modules to two state-of-the-art frameworks including PVN3D [9] and DenseFusion [38], and evaluate their performance on YCB-Video. Note that PVN3D cannot utilize PRN directly, since segmentation is required on the whole scene while PRN focuses on specific objects. We thus only evaluate the effect of MMF-GCN on PVN3D. As shown in Table 6, PRN promotes the ADD-S of DenseFusion by 1%, and a similar improvement can be observed when applying MMF-GCN. The results indicate that PRN and MMF-GCN benefit other frameworks for 6D pose estimation.

The influence of segmentation. As in Fig. 2, our framework introduces RGB-based segmentation to extract foreground objects, while PoseCNN [40], DenseFusion [38] and PVN3D [9] adopt different instance segmentation models. To validate the effect of segmentation, we replace the segmentation model in our framework by the counterparts used in PoseCNN and PVN3D as well as the ground-truth, and report their AUC and ADD-S<2cm metrics on YCB-Video. Similarly, we evaluate this factor on other frameworks, including PoseCNN, DenseFusion and PVN3D. As reported in Table 7, all these frameworks achieve the highest AUC and ADD-S<2cm using ground-truth, indicating that better segmentation boosts the estimation accuracy. Meanwhile, with segmentation alternatives, our framework consistently outperforms the others, showing that PR-GCN is superior, regardless of which segmentation model is used.

The effect of the multi-resolution regression loss on PRN. We finally validate the credit of the regularized multi-resolution regression loss $L_{prn}$. For comparison, we apply $L_{prn}$ on $\hat{P}_o$ only, denoted by PRN-SR, while the multi-resolution case is denoted by PRN-MR. We also report the result without using PRN (WO-PRN). As summarized in Table 8, adopting the single-resolution loss promotes the performance of our method. When the multi-resolution loss is added, ADD-S is further boosted to 95.8%, demonstrating its effectiveness. Moreover, we visualize the generated refined 3D points in Fig. 5, and the results clearly show the advantage of PRN in dealing with the incompleteness and noise, after adding the loss $L_{prn}$.

5. Conclusion

In this paper, we propose a novel approach, namely deep Graph Convolutional Networks with Point Refinement (PR-GCN), to RGB-D based 6D pose estimation. We develop a Point Refinement Network (PRN) to improve the quality of depth representation, together with a Multi-Modal Fusion Graph Convolution Network (MMF-GCN) to fully explore local geometry-aware inter-modality correlations for sufficient combination. Extensive experiments validate its superiority and the PRN and MMF-GCN modules.

Acknowledgment

This work is partly supported by the National Natural Science Foundation of China (No. 62022011), the Research Program of State Key Laboratory of Software Development Environment (SKLSDE-2021ZX-04), and the Fundamental Research Funds for the Central Universities.
References


