6D Object Pose Estimation With Color/Geometry Attention Fusion

Honglin Yuan[1] Remco C. Veltkamp[1]

***Abstract*— The 6D object pose is widely applied in robotic grasping, virtual reality and visual navigation. However, heavy occlusion, changing light conditions and cluttered scenes make this problem challenging. To address these issues, we propose a novel approach that effectively extracts color and depth features from RGB-D images considering the local and global geometric relationships. After that, we apply a graph attention mechanism to fully exploit representations between these features and then fuse them together to predict the 6D pose of a given object. The evaluation results indicate that our method significantly improves the accuracy of the estimated 6D pose and achieves the state-of-the-art performance on LineMOD, YCB-Video, and a new dataset. Ablation studies demonstrate the effect of our network modules.**

I. Introduction

6D pose estimation is crucial for augmented reality, virtual reality, robotic grasping and autonomous navigation [1]. However, estimating object poses is challenging due to the variety of objects in the real world. They have varying 3D shape and the quality of captured images from them are affected by sensor noise, changing lighting conditions and occlusion. With the emergence of cheap RGB-D sensors, the precision of 6D object pose estimation is improved for both rich and low textures objects [2]. Nonetheless, existing methods still have difficulty to meet the requirement of accurate 6D pose estimation for objects with reflective property, and heavy occlusion.

Previous methods with RGB-only images as input work by extracting and matching hand-crafted features, and then 6D pose is estimated by solving a Perspective-n-Point(PnP) problem. Such methods are often fast and robust to occlusion. However, they heavily rely on rich features and are unable to handle texture-less objects. Instead of relying on improving handcrafted features, we learn more robust features and semantic cues by applying deep learning models.

Taking the advantage of depth sensors, RGB-D based methods [3], [4] predict more accurate 6D pose of low-textured objects than RGB-only methods. Nevertheless, these algorithms require a time-consuming pose refinement step, such as iterative closest point (ICP) algorithm to improve pose accuracy.

Recent approaches [1], [5] introduce end-to-end deep learning networks to improve 6D pose estimation with the fused color and geometric feature. In order to extract geometric information from the depth map, they first transform the depth map to a point cloud and then operate on each point independently. However, these methods do not consider relationships between point pairs, which results in the loss of local features and the decreased accuracy of the estimated 6D pose. To address this challenge, we apply edge convolution which considers both local and global point structures to compute geometric features.

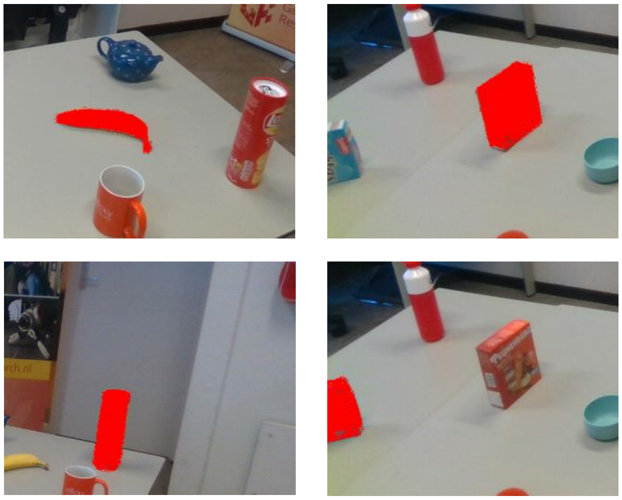


Fig. 1: Visualization of estimated poses by our method. Each 3D model is projected to the image plane with the estimated 6D pose.

Apart from discriminative geometric features, fusing color and geometric features is also important for improving the accuracy of estimated poses. Since these two types of features are defined in different spaces, fusing them is challenging. Existing approaches [1], [5] just concatenate these two kinds of features, which fail to fully exploit the correlation between them. Unlike Previous approaches, we introduce a graph attention based framework to effectively compute the hidden representations between visual and geometric features and then fuse them properly. To the best of our knowledge, this is the first work that uses the graph network to fuse color and geometric features for 6D pose prediction. In summary, we present two main contributions:

- Effectively extracting local and global geometric features from point clouds, which makes it robust to handle heavy occlusion, low texture and sensor noise for 6D object pose estimation.
- A new multi-feature fusion network improving 6D pose prediction performance, that applies a graph attention network (GAT) to fully exploit the relationship between visual and geometric features and compute hidden feature representations between these features.

[1]Honglin Yuan and [1]Remco C. Veltkamp are with the Department of Information and Computing Sciences, Utrecht University, 3584CC Utrecht, The Netherlands h.yuan@uu.nl, R.C.Veltkamp@uu.nl

We show results on a variety of objects (see Fig. 1), demonstrating that our proposed method provides accurate 6D object pose. Besides, our approach achieves state-of-the-art performance on popular benchmark datasets, including LineMOD [6] and YCB-Video [7] datasets.

II. Related work

6D pose estimation has been an active research area for a long time and a review of 6D pose estimation approaches can be found in [8]. Here we only discuss the most related approaches with our method.

Pose estimation based on RGB images. Traditional RGB-based methods establish the 2D-3D correspondences between 2D key points and 3D models either by extracting and matching local features or predicting 2D projections of predefined 3D key points. Based on these correspondences, 6D poses are estimated by solving PnP problems [9], [10]. Although these algorithms are effective and fast for rich texture objects, they have difficulty in handling low-textured or no-textured objects. Other methods [6], [11] use learning-based methods to directly estimate 6D object poses from color images. For example, PoseNet [11] and PoseCNN [6] directly regress to the 6D object pose by convolutional neural network based architectures from single RGB images. However, their predictions are sensitive to small errors due to the large search space and they require careful tuning hyper-parameters for the associated loss functions.

Pose estimation based on RGB-D images. A different class of approaches takes the advantage of depth sensors that provide rich information for texture-less objects. These methods [12], [13] extract 3D features from color-and-depth image pairs and then perform correspondence matching to predict 6D poses. Ipose [12] uses an encoder-decoder architecture to extract features from color image and then obtains the 2D-3D correspondences between the color image and the 3D model. Instead of predicting pose directly, the 6D pose is estimated by solving the PnP problem with the obtained correspondences and depth information.

On the other hand, recent methods [14], [1] use the fused RGB-D data to directly estimate the 6D pose. Michel et al. [14] fuse the RGB-D information in the early stage, where the depth information is treated as a fourth channel and concatenated with RGB channels. Alternative solutions including Densefusion [1] fuse the color and depth information in the later stage, which generate dense pixel-wise features to estimate poses. However, these methods fail to effectively exploit the fuse strategy between color and geometric information.

Graph-based features. Graph neural network(GNN) is a deep learning-based method to aggregate feature information from input data, especially suitable for data lying on irregular or non-Euclidean domains, such as point clouds. It has been successfully applied in many areas, such as semantic segmentation [15] and physics systems [16]. To effectively extract geometric information, we use graph based edge convolution to process the point cloud. Edge convolution (EdgeConv) [20] generates edge features that describe the relationships between a point and its neighbors. It avoids the fundamental limitation that leads to loss of local features produced by previous approaches.

Attention mechanisms have been used together with many neural network architectures that operate on regular and graph-structured data. Self-attention which is used to compute representations of sequence-based data attracts many attentions. It has been applied successfully in tasks such as machine translation [21] and object detection [15] on Euclidean domains. Veličković et al. [22] propose graph attention networks that operate on graph-structured data. They have achieved state-of-the-art results on transductive and inductive graph benchmarks. Inspired by this work, we propose a graph attention based architecture to fuse RGB-D data for 6D pose estimation.

III. Methodology

A. Overview

Our goal is to achieve accurate 6D pose estimation for objects with a wide range of sizes, shapes and textures using RGB-D images as input. Unlike previous algorithms designing new characteristics to improve the robustness of handcrafted features, we apply a CNN-based encoder-decoder architecture to learn features from color images.

Rather than directly extracting features from the depth map, we first convert it to a point cloud which contains rich geometric information using the camera intrinsic matrix. To process point cloud data, previous methods first convert it into regular grids by projecting 3D data into 2D images or splitting 3D data into 3D voxel grids. Then they process the transformed data using approaches designed for regular data. However, these approaches either introduce quantization artifacts or result in missing local features. To overcome these limitations, we use edge convolution [20] which can capture local geometric structure while maintaining permutation invariance.

Fusing features is crucial for high-precision pose estimation. To effectively fuse visual and geometric features, we introduce a graph attention based framework to exploit relationships between them, as opposed to prior works which just concatenates these features.

Fig. 2 shows the overall framework. We first perform semantic segmentation to extract the target object from color-and-depth image pairs (Fig. 2(b), section III-B). Next, we extract color and geometric features, separately, retaining the native structure of each data (Fig. 2(c), section III-B). We apply a CNN-based network to aggregate appearance information in the color image. To extract local and global geometric features from the depth map, we first convert the depth map to a point cloud and then build the local graph map for each point with the k-nearest neighbors algorithm (kNN). After that, the geometric features are computed by edge convolution on each local graph map. Furthermore, we fuse visual and geometric features with a graph attention based network (Fig. 2(d), section III-C). Finally, we train the network to predict the 6D pose for chosen pixels and

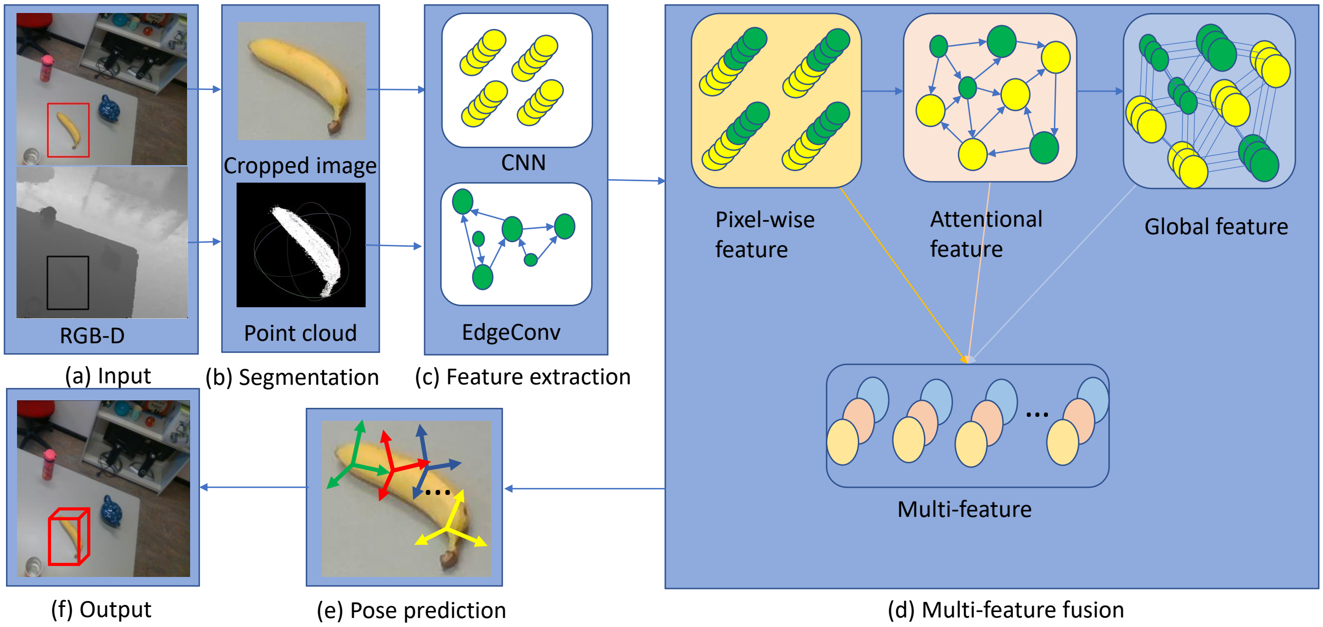


Fig. 2: Overview of our architecture: (a) The input are captured color-and-depth image pairs. (b) A semantic segmentation architecture is used to segment the target object. (c) The visual features are extracted by a CNN-based network from the color image and geometric representations are computed from the point cloud converted by its corresponding depth map. (d) A graph attention network is introduced to perform the fusion between color and geometric features. (e) The 6D object pose and its corresponding confidence score are predicted by the fused features, one pose per fused feature. (f) The pose with the highest confidence is chosen as the final pose.

then apply an iterative refinement method to obtain the final pose (Fig. 2(e), section III-D).

B. Semantic segmentation and feature extraction

Semantic segmentation. We detect objects in the input image using semantic segmentation from [6]. It generates a per-pixel segmentation map which classifies each pixel into a known object class. From the segmentation map, we get a 2D bounding box for the target object. Then we use the bounding box to crop the input color-and-depth image pairs.

Feature extraction. In order to effectively extract information from color and depth images, we process the cropped color-and-depth image pairs separately. This is because the color data can be represented in a grid-like structure, while the geometric information residing in the depth map is defined in a continuous vector space. The cropped color image is fed into a CNN-based encoder-decoder architecture to extract visual information. Specially, given a color image of size $H \times W \times 3$, the network generates a feature map of size $H \times W \times d_{rgb}$ which contains the d_{rgb}—dimensional hidden representation of each pixel in the color image.

To extract geometric features, we first project the cropped depth map to a point cloud based on the camera intrinsic matrix. Even though features learned from each point in the point cloud are able to encode the neighboring geometric structure of each point, such features suffer from sensor noises. In contrast, point-pair features extracted from multiple points, are robust to occlusion and noises. To effectively extract point-pair features between points, we build a graph map for each point using kNN (Fig. 3(b)). Then we use edge convolution [20] which applies convolution-like operations on the local graph to extract features. These features describe the relationships between a point and its neighbors (Fig. 3(c)).

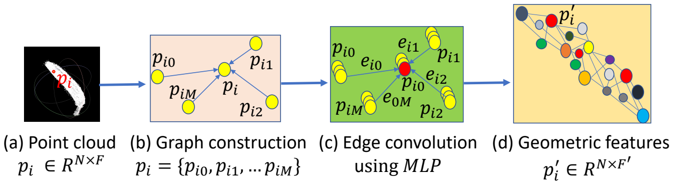


Fig. 3: Geometric feature extraction: (a) The input is a point cloud with N points. (b) Each point of the point cloud is clustered by kNN to produce graph maps. (c) Geometric information is extracted by edge convolution. (d) A new set of features are produced as the output.

The input to the edge convolution layer is a local graph with M points, denoted by $p_i = \{p_{i0}, p_{i1}, ..., p_{iM}\}$, $p_i \in R^{N \times F}$, N is the number of points in the point cloud, F is the dimension of each point. The edge feature is defined as $e_{i,j} = f(p_i, p_j - p_i)$, where $f : R^F \times R^F \rightarrow R^{F'}$ is parametric non-linear function parameterized by a learnable weight matrix. F' is the new dimension of each point. We compute the edge feature by applying a multi-layer perceptron (MLP) and the output , $p_i' \in R^{N \times F'}$ is shown in Fig. 3(d).

C. Multi-feature fusion

A simple fusion of color and geometric features is to directly concatenate them. However, that is not able to effectively exploit the relationships between these features for more accurate 6D pose estimation. The key idea of our multi-feature fusion is to apply GAT to compute the hidden representations of each feature by attending over its neighbors.

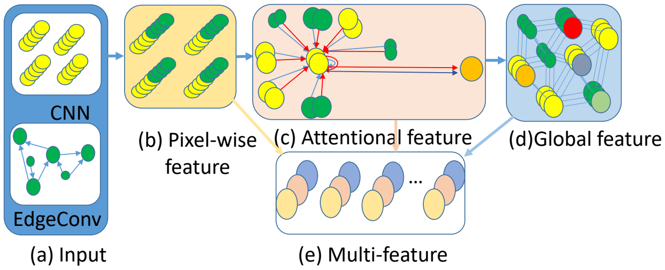


Fig. 4: Multiple feature fusion: (a) Color and geometric features are input. (b) We combine these two types of features to generate pixel-wise features. (c) The GAT is applied to compute attentional features. (d) The global features are generated by attentional features. (e) Features generated from (b), (c) and (d) are concatenated to produce multiple features.

Our multiple feature fusion procedure first concatenates color and geometric features as node features (4(b)) and then feed them to the graph attention network. The graph attention layer updates the node features. Concretely, the input nodes are first transformed by a linear transformation, parameterized by a weight matrix, $H \in R^{F' \times F}$, where F' is the number of new features in the node, to achieve a higher representation. Then a shared attention mechanism $GA : R^{F'} \times R^{F'} \rightarrow R^{2F'}$ is applied to the transformed node to compute attention coefficients $e_{i,j} = GA(Hx_i, Hx_j)$. The coefficient represents the importance of node j's features to node i.

In our experiments, we use a single-layer feedforward neutral network as the attention mechanism GA, parameterized by a set of learn-able parameters, $ga = \{ga_1, ga_2, ..., ga_N\}$, $ga_i \in R^{2F'}$, where N is the number of nodes. The LeakyReLU is used as the activation function and the softmax function is used to normalize the attention coefficients. The attention mechanism is expressed as:

$$ga_{ij} = softmax_j(LeakyReLU(ga^T[Hx_i||Hx_j])), \qquad (1)$$

where T is the transposition and $||$ is the concatenation operation.

After obtaining the learnable parameter for each node, it is multiplied with the node features by a nonlinearity, Θ, to produce the output node features. To obtain stable features, K attention mechanisms are implemented. Next, all the output features are concatenated:

$$x'_j = ||_{k=0}^{K} \Theta(\sum_{j \in N} ga_{ij}^k H^k x_j])), \qquad (2)$$

where the attention parameter, ga_{ij}^k, is computed by the k-th attention mechanism (ga^k), and H^k is the corresponding transformation matrix applied for the input node. In our experiment, K is set to be two, as shown in Fig. 4(c).

The resulting attentional features are fed into a CNN-based network to generate a global feature vector using maxing-pooling reduction function (Fig. 4(d)). Finally, we concatenate pixel-wise, attentional and global features together as our multi-fusion features(4(e)).

D. Pose estimation and refinement

Pose estimation. We predict the pixel-wise 6D object pose with our fusion features by MLP. We also predict a confidence score for the corresponding pose. It indicates the possibility of the corresponding pose to be the final one. During inference, we choose the pose with the highest score as the final predicted pose. The loss function is defined based on the Euclidean distance between points transformed by ground truth pose and those transformed by predicted pose:

$$l_i = \frac{1}{N} \sum_{j \in N} ||(Rx_j + t) - (R'_i x_j + t'_i||), \qquad (3)$$

where $[R'_i|t'_i]$, $i \in N$, and $[R|t]$ are the estimated and ground truth 6D pose respectively, x_j is the selected point from the 3D model and N is the number of selected points.

Pose refinement. The performance of 6D pose estimation can be further improved by iterative refinement. We adopt the refinement module from [1] to improve the pose accuracy. Concretely, the input of this step are color features computed from the cropped color image and geometric features computed from the new point cloud transformed by the predicted 6D pose. The idea behind this transformation is that the transformed point cloud implicitly encodes the predicted pose. Then the two kinds of features are fused and fed into the refinement network to predict a residual pose. The final pose is obtained by M iterations:

$$RT = [R_M|t_M].[R_{M-1}|t_{M-1}]...[R_0|t_0] \qquad (4)$$

We can train the pose refinement network and the main network together. In order to reduce the training time, we start the refinement network after the main network has converged.

IV. Experiments

A. Settings

Datasets. In our 3D object dataset [23], there are eight objects covering a variety of shapes, sizes and textures. We also evaluate our method on the LineMOD and YCB-Video datasets. LineMOD dataset contains of 13 texture-less objects and YCB-Video dataset have 21 objects of varying shapes and textures. We follow the same training and testing settings as prior learning based approaches [2], [1].

Evaluation metrics. The pose estimation performance is evaluated by ADD(-S) including the average distance metric (ADD) and the average closest point distance (ADD-S) [6].

TABLE I: Quantitative evaluation of the 6D pose (ADD(-S)) on the LineMOD dataset (objects with bold name are symmetric).

	ape	ben.	cam	can	cat	driller	duck	**eggbox**	**glue**	hole.	iron	lamp	phone	MEAN
DenseFusion	92.3	93.2	94.4	93.1	96.5	87.7	92.3	99.8	**100.0**	92.1	97.0	95.3	92.8	94.3
SSD-6D	65.0	80.0	78.0	86.0	70.0	73.0	66.0	**100.0**	**100.0**	49.0	78.0	73.0	79.0	77.0
Implicit+ICP	20.6	64.3	63.2	76.1	72.0	41.6	32.4	98.6	96.4	49.9	63.1	91.7	71.0	64.7
C/G-AF	**96.3**	**97.4**	**97.8**	**97.6**	**98.5**	**96.8**	**97.4**	99.8	**100.0**	**95.3**	**97.3**	**98.8**	**98.6**	**97.8**

TABLE II: Quantitative evaluation of the 6D pose (ADD(-S)) on the YCB-Video dataset (objects with bold name are symmetric).

	PoseCNN		DenseFusion		C/G-AF	
	AUC	ADD(-S) <2 cm	AUC	ADD(-S) <2 cm	AUC	ADD(-S) <2 cm
002_master_chef_can	68.1	51.1	73.3	72.3	**88.2**	**88.9**
003_cracker_box	83.4	73.3	**94.2**	**98.2**	93.5	94.2
004_sugar_box	**97.5**	99.5	96.5	**100.0**	96.7	**100.0**
005_tomato_soup_can	81.8	76.6	85.5	83.0	**93.0**	**95.8**
006_mustard_bottle	**98.0**	98.6	94.7	96.1	95.1	**98.9**
007_tuna_fish_can	83.9	72.1	81.9	62.2	**89.2**	**85.7**
008_pudding_box	**96.6**	**100.0**	93.2	98.6	95.6	99.3
009_gelatin_box	**98.1**	**100.0**	96.7	**100.0**	**98.1**	**100.0**
010_potted_meat_can	83.5	77.9	83.6	79.9	**87.5**	**84.6**
011_banana	91.9	88.1	83.7	88.4	**92.1**	**97.9**
019_pitcher_base	**96.9**	97.7	**96.9**	**100.0**	95.9	**100.0**
021_bleach_cleanser	**92.5**	**92.7**	89.7	90.8	90.0	89.5
024_bowl	81.0	54.9	89.5	95.1	**89.9**	**96.7**
025_mug	81.1	55.2	88.9	88.8	**93.5**	**97.1**
035_power_drill	**97.7**	92.2	92.7	**96.5**	89.9	91.1
036_wood_block	87.6	80.2	92.8	**100.0**	**93.4**	98.2
037_scissors	78.4	49.2	77.5	48.6	**91.9**	**89.3**
040_large_marker	85.3	87.2	93.0	**100.0**	**94.7**	99.8
051_large_clamp	75.2	74.9	72.5	**78.7**	**75.0**	78.2
052_extra_large_clamp	64.4	48.8	69.9	74.9	**73.9**	**76.8**
061_foam_brick	**97.2**	**100.0**	91.9	**100.0**	94.1	**100.0**
Mean	86.6	79.9	87.6	88.2	**91.0**	**93.4**

Given the ground truth rotation R and translation T and estimated rotation $\hat{R}$ and translation $\hat{T}$, the ADD computes the mean distances between all 3D model points x transformed by $[\hat{R}|\hat{T}]$ and $[R|T]$:

$$ADD = \frac{1}{N}\sum_{x\in S} ||(Rx+T) - (\hat{R}x+\hat{T})||, \tag{5}$$

where S is the set of 3D model points and N is the number of points.

The ADD-S is an ambiguity-invariant pose error metric which takes care of both symmetric and non-symmetric objects into an overall evaluation.

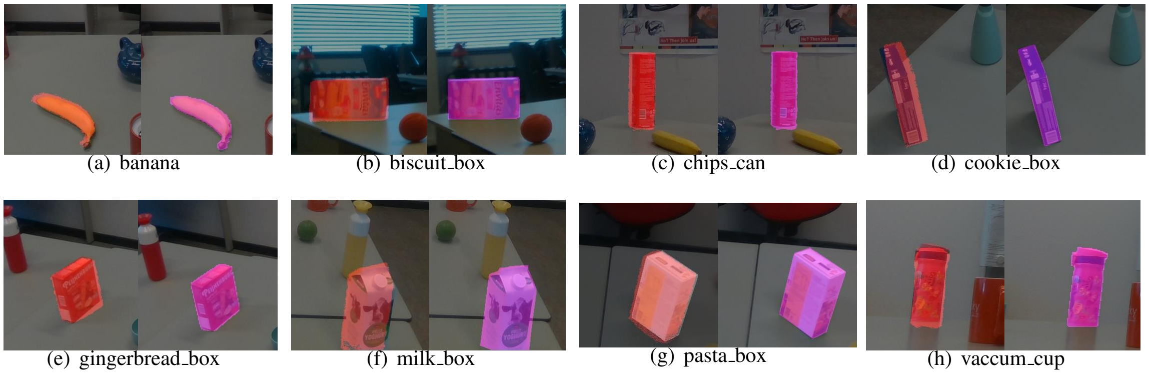
$$ADD\text{-}S = \frac{1}{N}\sum_{x_1\in S} \min_{x_2\in S} ||(Rx_1+T) - (\hat{R}x_2+\hat{T})|| \tag{6}$$

If the ADD(-S) is smaller than a threshold, we consider the estimated pose is correct. From that, we define a variable range of thresholds between $0.0cm$ to $10.cm$ following previous work and then compute the ADD(-S). Based on the two sets of values, we can plot the accuracy-threshold curve (AUC). Then we compute the area under the AUC as our another performance metric.

Implementation Details. We use the CNN-based network Resnet-18 [24] encoder followed by 4 up-sampling layers as decoder to extract color features. The edge convolution is a MLP with the number of layer neurons defined as $\{3, 64, 64, 64\}$. The graph is constructed using $k = 10$ nearest neighbors. A single-layer GAT model with 2 attention heads is used for the feature fusion. We implement the networks within the PyTorch framework and train our model using Adam optimizer and set the learning rate to 0.0001. Furthermore, we refine the pose predicted from the main work with 2 iterations.

B. Overall performance

The overall performance compared with other state-of-the-art approaches is shown in Table I and Table II. We use ADD(-S), including ADD for non-symmetric objects and ADD-S for symmetric objects, to measure the prediction on the LineMOD dataset. If ADD-S is smaller than $2cm$ which is the minimum tolerance for robot grippers, the predicted pose is considered to be correct. In Table I, we compare the percentage of ADD-(s) ($< 2cm$) with those of SSD-6D [4], Implicit+ICP [25] and DenseFusion [1]. The evaluation results compared with PoseCNN and Densefusion in terms of ADD-(S) and AUC on the YCB-Video dataset are shown in Table II. We can see that our proposed approach applying color/geometry attention fusion (C/G-AF) achieves the best performance on both datasets, which demonstrates that our fusion strategy is superior to those that do not exploit the relationship between color and geometric features. For the LineMOD dataset, our method outperforms Implicit+ICP and DenseFusion 33.1% and 3.5%, respectively.



(a) banana (b) biscuit_box (c) chips_can (d) cookie_box

(e) gingerbread_box (f) milk_box (g) pasta_box (h) vaccum_cup

Fig. 5: Examples of accuracy differences between DenseFusion and our approach on our dataset. The first image is the result of DenseFusion and the second image is the result of our approach.

TABLE III: Accuracy comparison with and without attention in terms of ADD(-S) on the LineMOD dataset (objects with bold name are symmetric).

	ape	ben.	cam	can	cat	driller	duck	**eggbox**	**glue**	hole.	iron	lamp	phone	MEAN
DenseFusion	80.5	83.2	78.5	88.1	89.3	79.9	77.9	99.6	99.1	79.5	93.2	93.5	89.2	87.1
C/G-AF(w/o attention)	85.3	88.5	88.7	88.9	90.1	87.5	89.3	98.2	98.9	84.9	88.6	90.4	89.9	89.9
C/G-AF	**93.8**	**95.4**	**95.6**	**95.8**	**96.5**	**92.7**	**95.7**	**99.7**	**100.0**	**91.3**	**95.9**	**96.6**	**96.8**	**95.8**

Furthermore, we also visualize the comparison results between DenseFusion and our method on our own dataset, as shown in Fig. 5. We can see that our approach provides accurate 6D pose, which indicates our method is more robust against occlusion and low texture objects.

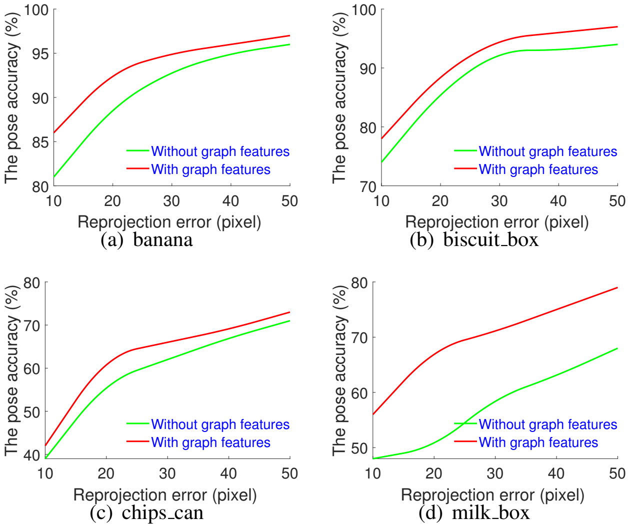
C. Ablation study

To verify the effectiveness of each module of our proposed network, we perform the ablation study on the LineMOD and our 3D object datasets.

Effectiveness of geometric feature extraction. By varying the reprojection error threshold for four objects on our dataset, we plot the accuracy-threshold curves, as shown in Fig. 6. It can be seen that the method using geometric features extracted from point pairs outperforms the approach which processes each point separately by a large margin, especially for low texture objects, such as the milk box.

Effectiveness of multi-feature fusion. Table III summarizes the comparison results with and without graph attention mechanism in terms of ADD(-S). As can be seen in Table III, compared with DenseFusion, the graph attention mechanism increases the accuracy significantly by 8.7%, and our approach predicts more accurate poses for symmetric objects, like glue. Compared with our method without multi-feature fusion module, the performance is also increased by a large margin (5.9%).

Robustness against occlusion. In Fig. 7 we compare our approach with DenseFusion, PointFusion and poseCNN + ICP in terms of the robustness against occlusion. We first calculate the visible surface ratio of each sampled object when projected to the image plane. Then we calculate the number of successful predictions among all the test frames. If the ADD(-S) is smaller than $2cm$, we consider the prediction is correct. In detail, we sample a set of points from the 3D object model and project these points to the image plane to synthesize a depth image using the ground truth 6D pose and camera intrinsic matrix. Next, we compare the pixel value in the synthesized depth image with the ground truth depth image. If the calculated pixel value is smaller than ground truth value, we consider its corresponding 3D point is invisible. This is because only the front-most pixels are shown in the depth image. After that, we calculate the number of invisible points from our sampled points and then obtain the invisible ratio. As shown in Fig. 7, our approach performs best among these methods. The increasing invisible ratio does not reduce our method's performance significantly,



(a) banana (b) biscuit_box

(c) chips_can (d) milk_box

Fig. 6: The accuracy-threshold curves (AUC) generated by reprojection error on our dataset.

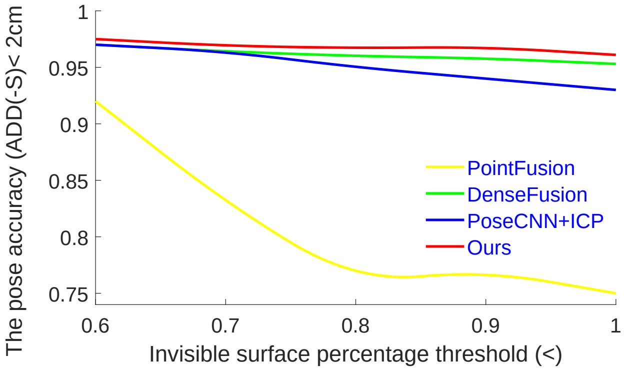


Fig. 7: Accuracy comparison with different degree of occlusion.

while PoseCNN degrades greatly.

V. Conclusion

In this paper, we propose a novel 6D object pose estimation framework that first extracts discriminative color and geometric features from RGB-D images. We then fuse these features based on a graph attention mechanism to predict the object pose. Experimental results have demonstrated that the feature extraction and fusion modules can increase the overall accuracy of estimated 6D poses, and the proposed approach can be used for robot grasping tasks. However, some limitations are worth noting. Although our method is robust to varying shape objects, when the object is under changing light conditions, our method still fails to predict the accurate pose. It would be interesting to explore more efficient approaches to estimate the 6D poses of objects that are under more challenging conditions in the future.

References

[1] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6D object pose estimation by iterative dense fusion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3343–3352.

[2] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 292–301.

[3] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6d object pose estimation using 3d object coordinates," in *European conference on computer vision*. Springer, 2014, pp. 536–551.

[4] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1521–1529.

[5] Z. Xu, K. Chen, and K. Jia, "W-PoseNet: Dense correspondence regularized pixel pair pose regression," *arXiv preprint arXiv:1912.11888*, 2019.

[6] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.

[7] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *2011 international conference on computer vision*. IEEE, 2011, pp. 858–865.

[8] C. Sahin, G. Garcia-Hernando, J. Sock, and T.-K. Kim, "A review on object pose recovery: From 3D bounding box detectors to full 6D pose estimators," *Image and Vision Computing*, p. 103898, 2020.

[9] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018.

[10] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis, "6-dof object pose from semantic keypoints," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2011–2018.

[11] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2938–2946.

[12] O. H. Jafari, S. K. Mustikovela, K. Pertsch, E. Brachmann, and C. Rother, "ipose: instance-aware 6D pose estimation of partly occluded objects," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 477–492.

[13] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, "Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation," in *European conference on computer vision*. Springer, 2016, pp. 205–220.

[14] F. Michel, A. Kirillov, E. Brachmann, A. Krull, S. Gumhold, B. Savchynskyy, and C. Rother, "Global hypothesis generation for 6D object pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 462–471.

[15] X. Liang, L. Lin, X. Shen, J. Feng, S. Yan, and E. P. Xing, "Interpretable structure-evolving lstm," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1010–1019.

[16] N. Watters, D. Zoran, T. Weber, P. Battaglia, R. Pascanu, and A. Tacchetti, "Visual interaction networks: Learning a physics simulator from video," in *Advances in neural information processing systems*, 2017, pp. 4539–4547.

[17] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, 2016, pp. 3844–3852.

[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[19] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[20] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, pp. 1–12, 2019.

[21] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[22] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[23] "3D object datasets," https://yhldrf.github.io/Datasets.github.io/, accessed: 2020-05-31.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[25] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3d orientation learning for 6d object detection from rgb images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 699–715.