Combined Holistic and Local Patches for Recovering 6D Object Pose

Haoruo Zhang
Shanghai Jiao Tong University
Shanghai, P.R.China
zhr@sjtu.edu.cn

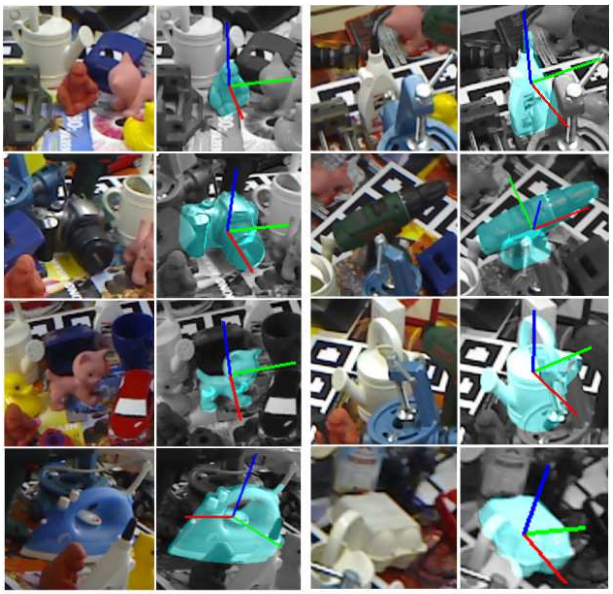
Qixin Cao
Shanghai Jiao Tong University
Shanghai, P.R.China
qxcao@sjtu.edu.cn

Abstract

We present a novel method for recovering 6D object pose in RGB-D images. By contrast with recent holistic or local patch-based method, we combine holistic patches and local patches together to fulfil this task. Our method has three stages, including holistic patch classification, local patch regression and fine 6D pose estimation. In the first stage, we apply a simple Convolutional Neural Network (CNN) to classify all the sampled holistic patches from the scene image. After that, the candidate region of target object can be segmented. In the second stage, as proposed in Doumanoglou et al. [16] and Kehl et al. [17], a Convolutional Autoencoder (CAE) is employed to extract condensed local patch feature, and coarse 6D object pose can be estimated by the regression of feature voting. Finally, we apply Particle Swarm Optimization (PSO) to refine 6D object pose. Our method is evaluated on the LINEMOD dataset [5] and the Occlusion dataset [10, 5], and compared with the state-of-the-art on the same sequences. Experimental results show that our method has high precision and good performance under foreground occlusion and background clutter conditions.

1. Introduction

The task of recovering 6D object pose has gained much more focus, because of its application in augmented reality and robotic intelligent manipulation to name but a few. Although it is a well-studied problem in computer vision, there are still remain many challenges such as foreground occlusions, background clutter, multi-instance objects and large scale and pose changes. As for sufficiently textured object, several methods based on sparse feature key-point matching [1, 2, 3, 4] demonstrate good results. Otherwise, objects have little texture and common local appearance descriptors are not discriminative enough to provide reliable correspondences for key-point matching. Recently, Hinterstoisser et al. [5, 6, 7] and related [8, 9] presented holistic template-based methods which can deal with 6D pose estimation of texture-less objects in a heavily cluttered scene. But, these methods suffer in the condition of foreground occlusions. In order to deal with above issues, several dense local feature-based methods like Brachmann et al. [10, 11] and Tejani et al. [12] employed a generalized Random Forest classifier and some simple pixel-based features. Although these methods has been proven to robust towards foreground occlusion, manually designed features are difficult to make discriminative for various objects. In recent years, much impressive performance boost in computer vision has been brought by convolutional neural networks (CNNs), which can automatically learn discriminating features from raw image. For instance, [13, 14, 15] mainly utilized CNNs to extract features and [15] trained CNNs to directly regress 6D object pose. These methods have moderate accuracy and cannot be directly



(a) (b)

Figure 1. Some results (zoomed view) of recovering 6D object pose on (a) the LINEMOD dataset [5], (b) the Occlusion dataset. Each detected object is augmented with its 3D model and its coordinate system. The background is kept in gray for better visibility.

applied to recovering 6D object pose without fine 6D pose estimation. Meanwhile, these holistic methods don't have enough robustness for occlusions. Next, Doumanoglou et al. [16] and Kehl et al. [17] presented local patches-based methods which trained autoencoder (AE) or convolutional autoencoder (CAE) to extract discriminating features from local patches and used these features to create 6D pose hypotheses. Although features of local patches can perform reliable 6D object detection and pose estimation under foreground occlusions, 2D-3D correspondences from local patches would still have much more noise.

In this paper, we propose a novel method which combines holistic patches classification, local patches regression framework and fine 6D pose estimation. As shown in Figure 1, the proposed method performs well on the public dataset. The input of proposed method is RGB-D image provided by a consumer-level RGB-D sensor. The **main contribution** of our work consists of three parts: Firstly, we generate normalized holistic RGB patches, apply a CNN to classify these patches and roughly locate object in 2D image. In this way, candidate region can be segmented from the scene image. Secondly, we apply local RGB-D patches regression framework to recover 6D object pose roughly after candidate region segmentation. Features of local patches are extracted by CAE. Thirdly, we employ iterative optimization algorithm for fine 6D pose estimation after roughly 6D object pose regression. Experiments show that the proposed method has good performance for objects in some public datasets.

The paper is organized as follows. Section 2 provides an overview of related work, and the proposed method is described in section 3. In section 4, we present evaluation of our method compared to the state-of-the-art on three public datasets. Finally, we conclude the paper in section 5.

2. Related Work

Recovering 6D object pose has been a long-standing and intensely researched activity in the field of computer vision and intelligent robotic application. Sparse feature key-point matching methods [1, 2, 3, 4] perform well on very textured objects. However, when applied to texture-less objects, common local feature descriptors are no longer discriminative enough to provide reliable correspondences. Therefore, many recent methods are presented, which don't employ any sparse feature key-points, including template-based methods [6, 8, 19, 20, 21], shape-based methods [22, 23, 24, 25, 26], dense feature-based methods [27, 28, 10, 11, 12, 13, 29], sparse feature-based methods [30, 16, 17] and direct regression methods [14, 15]. In consideration of the type of feature, these methods can also be divided into two main categories which are holistic methods and local methods.

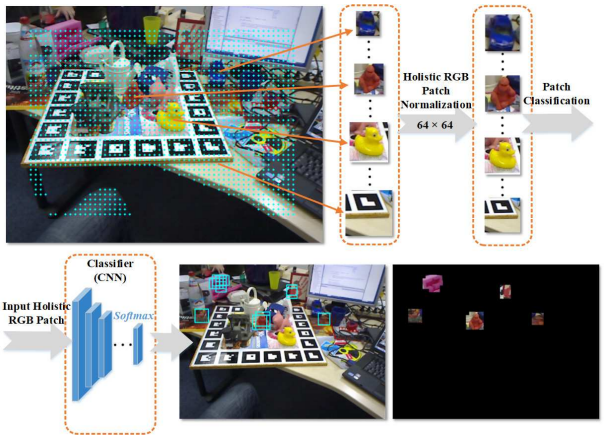


Figure 2. The brief pipeline of holistic patch classification.

As for holistic methods, 6D pose of the target object is predicted from its holistic appearance. Template-based methods mainly detect objects in the scene by employing holistic templates extracted from rendered views of 3D models. In Hinterstoisser et al. [5, 6, 7], each template is presented as one 6D pose of object and it consists of several quantized color gradient and surface normal. This method performed robust object detection in the clutter background. Recently, Cai et al. [19], Hodaň et al. [20], Zhang and Cao [21] employed cascade-type and hash-coded voting framework to optimize template matching and improved the accuracy of 6D pose estimation. In addition, Rusu et al. [23, 24] presented FPFH and VFH descriptors which are extracted from holistic point cloud model of the target object. Wohlhart et al. [15] applied deep network to directly regress 6D object pose from holistic patches, and it does hint towards replacing hand-crafted features with auto-learned ones for this task. Although these methods have high recognition rate in the clutter background, their recall drops quickly with partial foreground occlusions.

On the other hand, local methods mainly employ various discriminating local feature to regress 6D object pose. Brachmann et al. [10, 11] use a random forest to obtain pixelwise dense predictions. The split function in random forest is based on some simple pixel comparisons. Each decision tree in forest is trained to jointly predict where it is located in object coordinate system, which object the pixel belongs to. Each pixel in the scene can make a 3D continuous prediction about only its local correspondence to a 3D model. Then, the 6D object pose can be estimated using a PnP algorithm from these 2D-3D correspondences. Drost et al. [25] and Choi et al. [26] proposed oriented point pair feature which is also a kind of local feature. In these methods, several pairs of points is sampled from the scene and they are used to vote for 6D pose hypotheses of corresponding object. Doumanoglou et al. [16] and Kehl et al. [17] used local patch feature which can be extracted by

AE or CAE. After feature generation, Doumanoglou et al. [16] trained a Hough Forest by utilizing learnt features, and determined object classes and poses by utilizing 6D Hough voting. In Kehl et al. [17], local patch features of scene were matched against a codebook of synthetic model view patches and cast 6D object votes. Although these local patch-based methods are robust to occlusions, clutter background will lead to much noise for 6D object voting.

In summary, to the best of our knowledge, we are among the first to focus on the approach of combining holistic patch classification with local patch regression. The accuracy of the proposed method will be compared with the state-of-the-art, as shown in experiments.

3. Proposed Method

In this section, we will give a detailed description of the proposed method. Firstly, we will describe how to generate normalized holistic RGB patches. The model of holistic patches classifier is a CNN, which will be discussed subsequently. After that, candidate region segmentation will be described. Secondly, we will give a description of local patch feature regression in the segmentation image. Finally, we will describe how to recovering optimal 6D object pose and present our iterative optimization algorithm for fine 6D pose estimation.

3.1. Holistic Patch Classification

In our method, the aim of holistic patch classification is to roughly localize the target object in 2D image. It has three parts, including normalized holistic RGB patch generation, patch classification and candidate region segmentation. The brief pipeline is shown in Figure 2.

Recently, Zhang and Cao [21] generated normalized RGB-D patch, in order to reduce the number of template. In this method, these patches are all scale-independent and normalized by depth value check. Due to the input shape of classification network is fixed, the proposed method also employed scale-independent patches. As shown in Fig. 2, a list of candidate windows is extracted from a RGB-D scene image by using sliding window. In the proposed method, the holistic patch consists of three color channels. The number of windows is related by step size (e.g. 8 pixels). In order to ensure that candidate windows have the same scale, we take depth value at the window's center point and the patch pixel size is computed by the relation as follows:

$$L_w = f \cdot (s / z), \tag{1}$$

where L_w is the pixel size of candidate window. All holistic patches have the same metric size s, which is related with the bounding box size of target object. Therefore, different object has different metric size, some details about it will be

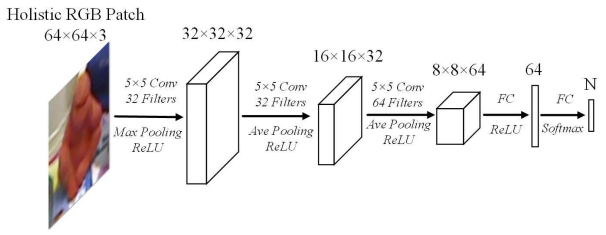


Figure 3. The architecture of holistic patch classification network.

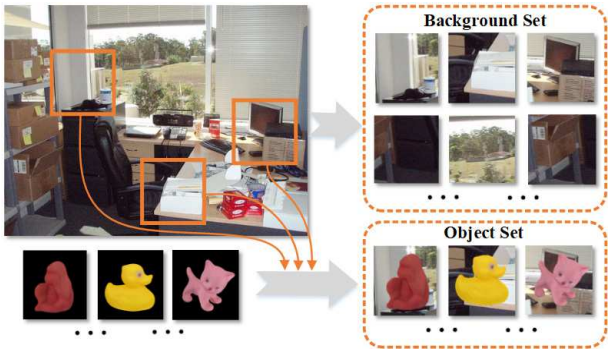


Figure 4. The training set generation for the classification task.

discussed in the section 4. In addition, f denotes the focal length of the camera, and z is the depth value at the window's center point. In this way, each holistic patch has the same scale but different pixel size. We normalize these patches with the same resolution ratio (e.g. 64×64), which is related to the input shape of classification network.

Instead of utilizing complex classification network like VGG [31] and GoogLeNet [32], we employ a much simpler architecture to classify holistic patches since these patches has low resolution. Beyond that, simple architecture has low computational cost and memory footprint. The architecture of classification network is shown in Figure 3. Because the number of target objects in some public datasets is about 10 (e.g. the LINEMOD dataset [5] is 15, the Occlusion dataset [10, 5] is 8), this kind of simple network can also work for classification.

As for network training, we need to generate several rendered images with different orientation poses to build training set at first, which is the same with Zhang and Cao [21]. In this section, we will describe the process of building training set for objects in the dataset of Hinterstoisser et al. [5]. Due to this dataset has 15 objects, the node number of softmax layer in the network is set as 16 (15 objects + 1 background). In order to improve the generalization of classification network, we need to augment the training set by using general background, as shown in Figure 4. As for background images, we randomly sample 64×64 patch from images in PASCAL VOC dataset [33]. Meanwhile, we combine random background patch with normalized rendered object image

to generate final training image. This effort can make the result of classification not be influenced by the scene. As shown in Figure 4, the foreground of final training image is the appearance of object, and the background is random patch in PASCAL VOC dataset.

In the testing phase, all the normalized holistic RGB patches will be fed into the classification network. The softmax layer of classification network can give us the probability distribution of categories. In this way, each patch can obtain the category label which has the maximum of probability. In this paper, we mainly focus on recovering 6D pose of a single object in the scene. Therefore, we will segment the candidate region of one target object from the scene image next. As shown in Figure 2, candidate region consists of holistic patches associated with target label, and we will use the segmentation image to regress the 6D pose of target object as described below.

3.2. Local Patch Regression

Recently, there are some methods which can directly predict 6D object pose in the scene, such as [14, 15]. However, these methods all have a moderate accuracy and low robustness to occlusions. Besides, local methods can keep the high robustness to foreground occlusions inherently. As inspired by [30, 16, 17], we apply local patch regression framework to recover the 6D pose of target object in the segmentation image. In the proposed method, local patch regression can be divided into three parts, including local patch feature representation, feature voting, and 6D pose regression. Figure 5 shows the overview of this framework.

Above all, we need to train the network to produce discriminative local patch features. The aim of our network is to learn a mapping from high-dimensional local patch space to a much lower feature space, and we employ a convolutional autoencoder (CAE) to extract local patch features. The cost function of this network is a reconstruction error between the input local patch p and output p_o while the inner-most layer condenses the data into $f(p)$ values. The network architecture is shown in Figure 6. In order to keep excellent abilities of learning and generalization, we teach the network regression on a great number of input local patches by randomly sampling in the dataset of Doumanoglou at el. [16] instead of Hinterstoisser et al. [5] (Our method is evaluated in the LINEMOD dataset [5] and the Occlusion dataset [10, 5]). As shown in Figure 6, each sampling local patch is normalized by depth value check which is the same with the part of holistic patch classification. As for local patch normalization, we de-mean the depth value and clamp them to a fixed metric size (e.g. 0.05 m) along x, y, and z axis. Then, we normalize color and depth value to [-1, 1] and resize local patched to 32×32×4. After network training, we need to build the

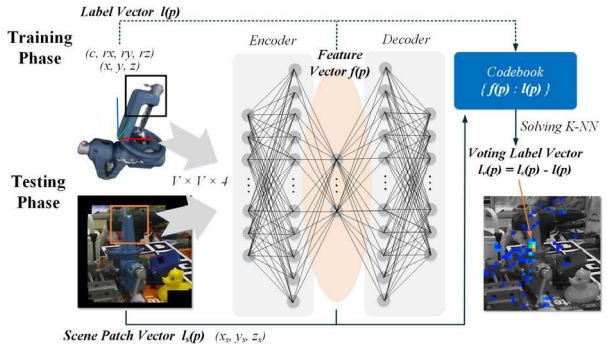


Figure 5. The general framework of local patch regression.

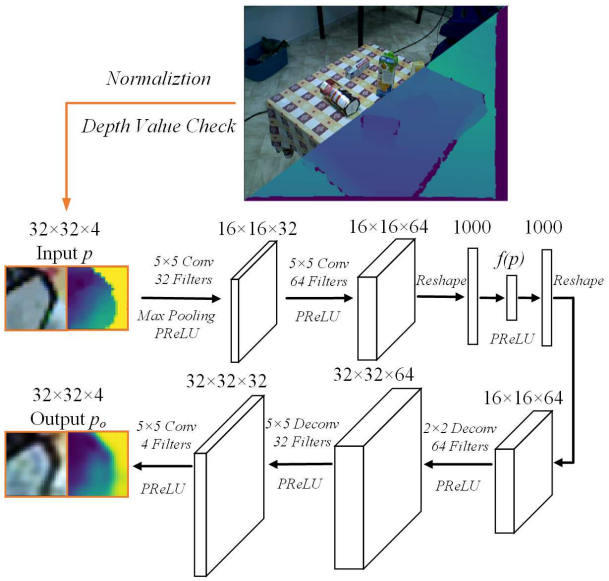


Figure 6. The architecture of convolutional autoencoder in our method. We use sampled RGB-D patches from the dataset of Doumanoglou at el. [16] to train this network and minimize a reconstruction error between input and output patches.

codebook of target object. As shown in Figure 5, we sample normalized local RGB-D patches from each synthetic view of target object and feed them into the network. Each patch p will be associated with a condensed feature vector $f(p)$. Meanwhile, each patch p has its label vector $l(p)$ which consists of patch category c, the rotational transformation (rx, ry, rz) from camera coordinate frame to object coordinate frame, and the patch 3D center point offset to the object centroid (x, y, z) under object coordinate frame. The codebook is made up of these feature vectors and associated label vectors.

As for feature voting, each sampled scene local patch p is associated with scene patch vector $l_s(p) = (x_s, y_s, z_s)$. These scene local patches will be fed into the network and condensed features $f(p)$ will be computed. Then, we can get k nearest neighbors (e.g. $k = 3$) from the codebook by

comparing scene patch features with original patch features, and Euclidean Distance is employed as feature distance. Each neighbor casts a global vote with the voting label vector $l_v(p) = l_s(p) - l(p)$, where $l(p)$ is one of k nearest neighbors and the rotational transformation is invariant. In our method, we firstly train features in the codebook and employ kd-tree structure to rebuild the codebook. In this way, nearest neighbor search can be very fast. Additionally, we define a threshold T_h on the nearest neighbor distance, so that k nearest neighbors will only cast vote if they have an enough confidence. It can decrease a great number of incorrect vote and reduce the noise sensitivity.

After feature voting, 6D pose regression need to be used and it will lead to a coarse 6D pose estimation of target object. In this stage, a series of crowded votes $l_v(p) = (x_v, y_v, z_v, rx, ry, rz)$ will be divided into two parts. One part represents translational transformation (x_v, y_v, z_v) and the other one represents rotational transformation (rx, ry, rz). In our method, we employ mean shift algorithm with a flat kernel to regress centers of translation and rotation respectively. In practical, the kernel size for translation regression is set as 1 cm and the other one is set as 5 degree. Additionally, rotational transformation need to be converted into quaternion space because of its differentiability. Mean shift algorithm can generate one center point for each cluster. In our method, we will remain the center point which has the most votes. Then, translational regression center and rotational regression center can make up for the coarse 6D pose estimation. Because there still remain some noise in feature voting, the 6D pose regression is not precise enough. Therefore, a stage of fine 6D pose estimation is needed.

3.3. Fine 6D Pose Estimation

In the proposed method, our final goal is to recover accuracy 6D pose of target object in the scene. In this stage, the input is a coarse 6D pose estimation, a point cloud model of the target object. As inspired by [20, 21, 34], we formulate the fine 6D pose estimation as an optimization problem. During the process of optimization, many candidate poses will be produced and each pose can generate a synthesized rendered depth image. Then, a cost function is needed to express the error of 6D pose estimation, and we define the cost by comparing synthesized rendered depth image with the observed depth image. In our method, Particle Swarm Optimization (PSO) is employed to refine 6D object pose, which is proven to be an efficient approach to optimize the cost function and find the pose whose rendered depth image is the most similar to the observed one.

The cost function is made up of two components, including depth value cost and depth edgelet cost, which can be formulated as follows:

$$C = cost(S, O) = \varepsilon_d C_d(S, O) + \varepsilon_e C_e(S, O), \quad (2)$$

where S represents synthesized rendered depth image and O represents observed depth image, and the component $C_d(S, O)$ will punish deviations of depth value between the synthesized rendered depth image and the observed depth image. The component $C_e(S, O)$ punishes deviations of depth edgelet. ε_d and ε_e are coefficients about the reliability of different components. In practical, depth value is more significant than the others. ε_d is set as 0.8 and ε_d is set as 0.2. The depth value component is directly compared between S and O at each pair of pixels and the depth edgelet component is based on distance between the depth edgelet in S and the closest depth edgelet in O. The definitions of these components are the same with [21]. Then, in order to find the optimal solution for this optimization problem, several candidate poses $\{R_c, t_c\}$ will be generated for each iteration, which are parameterized relative to the initial pose $\{R_0, t_0\}$ by using a series of relative rotations R_r and translations t_r. As shown in formula (3), the relative rotation R_r is made up of primitive rotations about X, Y, and Z axis. α, β, and γ are rotation angles around X, Y, Z axis respectively.

$$\begin{cases} R_c = R_0 \cdot R_r = R_0 \cdot R_x(\alpha) \cdot R_y(\beta) \cdot R_z(\gamma) \\ t_c = R_0 \cdot t_r + t_0 = R_0 \cdot (x, y, z)^T + t_0 \end{cases}, \quad (3)$$

Compared to gradient-based optimization, PSO does not require the derivatives of the cost function and has a wider basin of convergence, exhibiting better robustness to local minima [34]. As for fine 6D pose estimation problem, each candidate poses can be seen as candidate particles, and the search space is constrained in a 6D neighborhood of the initial 6D object pose. The state of each particle (candidate pose) will be updated after the completion of each iteration until there is no particle with less score of cost function. In the final stage, if one of particles (candidate poses) can get minimum score of cost function than all other particles (including updated particles and original particles) at certain iteration, particles won't be updated and the optimization process will stop.

4. Experiments and Discussion

In this section, we evaluate our method to other state-of-the-art methods for public datasets, including the LINEMOD dataset [5] and the Occlusion dataset [10, 5]. Additionally, we will discuss about the influence of several parameters in our methods. Owing to combine holistic and local patches, the proposed method could be called as HLP. The implementation of deep networks in HLP method is based on Caffe [18].

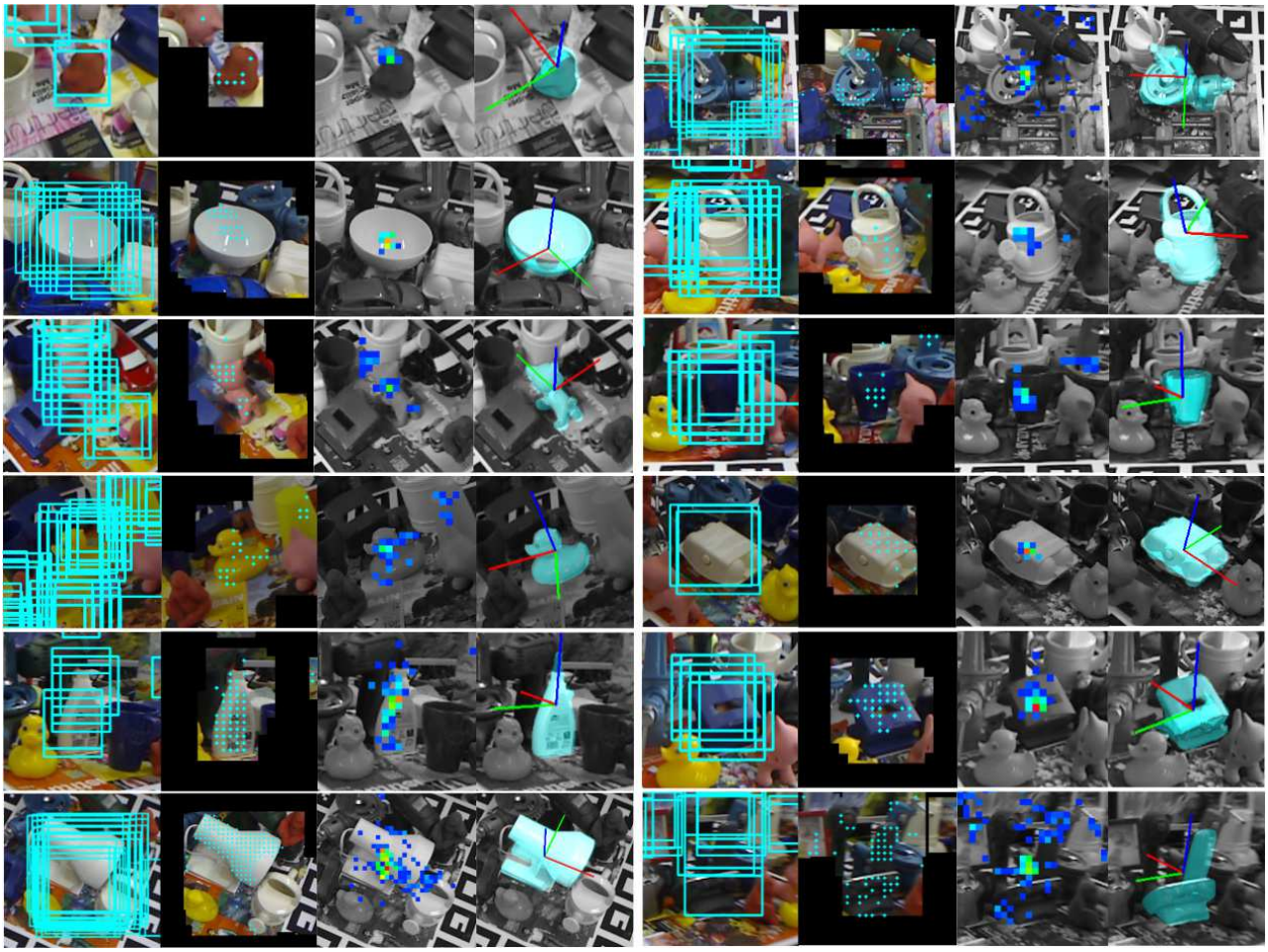


Figure 7. Some results of our method for the dataset of LINEMOD [5]. Each detection process has four parts in this image. From left to right, the first part is candidate region segmentation by using holistic patches. The second part shows sampled scene points for local patch regression, where threshold check for *k*-nearest neighbor search is adopted. The sampling step is fixed as 8 pixels in experiments. The third part shows results of feature regression. For better visibility, we subdivide the image plane into several 2D grid (cell size is 8×8 pixels) and cast each feature vote into cells. Each cell is colored according to the number of projected votes. The last part shows the results of recovering 6D object pose, where the background is also kept in gray for better visibility.

4.1. LINEMOD Dataset

This dataset provides colored 3D mesh models of 15 texture-less objects. In our experiments, we employ electrostatic-based rendered image creation method to generate a series of rendered object RGB-D images, and the detail can be found in [21]. In practical, each target object has 3600 rendered RGB-D images which can cover much more 6D poses.

As for the first stage of HLP, we need to build training set for holistic patch classification. The training set has 16 categories (15 objects + 1 background) and consists of numerous holistic RGB patches. As described in section 3.1, different object has different metric size of holistic patch metric size, such as 8cm for "Ape", 20cm for "Driller", and 18cm for "Bowl" and so on. In the stage of local patch regression, we also employ different metric size of patch for different object instead of the fixed one in [17]. It is suitable that small object has small metric size of local patch and vice versa, such as 3cm for "Ape", 8cm for "Driller", and 4cm for "Cam" and so on. In addition, the dimension of local patch feature is fixed as 200. According to several experiments, it has been proven that these parameters are the optimum value for local patch regression. After local patch feature extraction, features in the codebook need to be trained and the codebook will be converted into kd-tree structure. The number of trees is set as 4 in our method. In the third stage, there are much more parameters, including number of candidate poses N, the range of rotation angle θ and displacement τ. Some details will be described in section 4.3. In practical, the number of

Seq.	Method				
	[5]	[17]	[20]	[21]	HLP
Ape	0.958	0.969	0.939	0.963	**0.970**
B. Vise	0.987	0.941	**0.998**	0.904	0.951
Bowl	0.999	0.999	0.988	-	0.999
Cam	0.975	**0.977**	0.955	0.913	0.953
Can	0.954	0.952	0.959	**0.982**	0.969
Cat	**0.993**	0.974	0.982	0.964	0.979
Cup	0.971	**0.996**	0.995	-	0.995
Driller	0.936	**0.962**	0.941	0.952	0.960
Duck	0.959	0.973	0.943	0.918	**0.977**
Egg B.	0.998	0.999	**1.000**	0.989	0.999
Glue	0.918	0.786	**0.980**	0.946	0.830
Hole P.	0.959	0.968	0.880	**0.978**	0.972
Iron	0.975	0.987	0.970	0.988	**0.989**
Lamp	**0.977**	0.962	0.888	0.914	0.949
Phone	0.933	0.928	0.894	-	0.903
Mean	**0.966**	0.958	0.954	0.951	0.960
std.	**0.023**	0.050	0.039	0.105	0.042

Table 1. Recognition rate of each target object (sequence) by using our method and the other state-of-the-art methods.

candidate poses N is set as 25, and θ, τ are set as 5° and 5 mm respectively. Finally, in order to compare our method with the state-of-the-art, we need to employ consistent evaluation metric. Up to now, there are three basic types of evaluation metric for recovering 6D object pose, including average distance of model points, translational and rotational error, and complement over union. These evaluation metrics have different application scenarios, such as robotic manipulation and augmented reality. In our experiments, we take average distance of model points as standard 6D pose evaluation metric, which is the same with [5, 17, 20, 21]. The recognition rates for different methods are shown in Table 1. Figure 7 shows results of our method for the dataset of LINEMOD [5].

4.2. Occlusion Dataset

The dataset was created by Brachmann et al. [10], which is also standard public dataset in ICCV2015 Occluded Object Challenge. Target objects in this dataset are the same with LINEMOD dataset [5], and testing images are also selected from LINEMOD dataset [5]. The partial occlusions in these testing images make it significantly more difficult to recover 6D object pose.

We still use pre-trained network to accomplish holistic patch classification and local patch regression, because of the same target objects. In this dataset, there are eight target objects, including “Ape”, “Can”, “Cat”, “Driller”, “Duck”, “Eggbox”, “Glue”, and “Holepuncher”. Brachmann et al. [10] employed three different evaluation metrics to test their method and provided related recognition rate. These evaluation metrics are “AD”, “5cm, 5deg”, and “IOU”. “AD” is the average distance between all vertices in the 3D model of the part in the estimated pose and the ground truth pose. An estimated pose is considered correct, when the average distance is below 10% of the object diameter. “5cm, 5deg” is presented by Shotton et al. [35]. It denotes that an estimated pose is considered correct when the translational error is below 5cm and the rotational error is below 5deg. “IOU” is to calculate the intersection over union of the bounding boxes between estimated pose and ground truth. An estimated pose is considered correct when the IOU value is above a threshold of 0.5. Recently, there is also some amazing work about recovering 6D object pose for the Occlusion Dataset, such as [36] and [37]. Table 2 shows recognition rate comparison among these methods and our method (HLP). Some qualitative results of HLP method in this dataset are also shown in Figure 8.

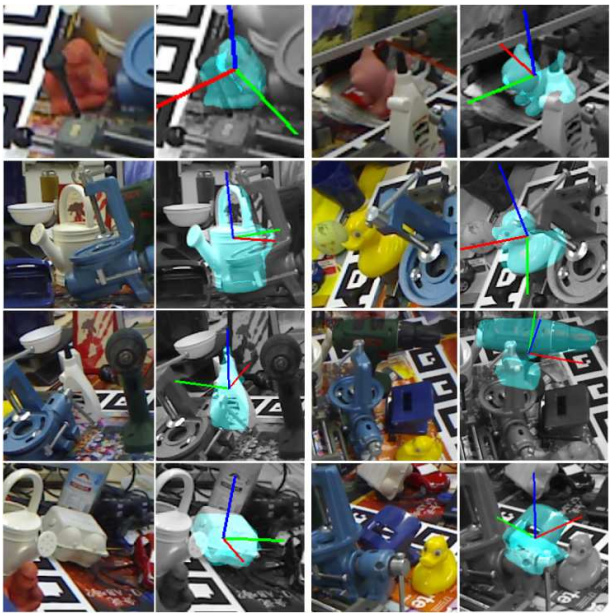


Figure 8. Some results of our method for the Occlusion dataset.

Method	AD	5cm, 5deg	IOU
[10]	56.6%	22.7%	62.0%
[5]	54.4%	-	-
[29]	70.3%	-	-
[36]	76.2%	-	-
[37]	**76.7%**	-	-
HLP	63.2%	**36.2%**	**79.9%**

Table 2. Overall recognition rate comparison.

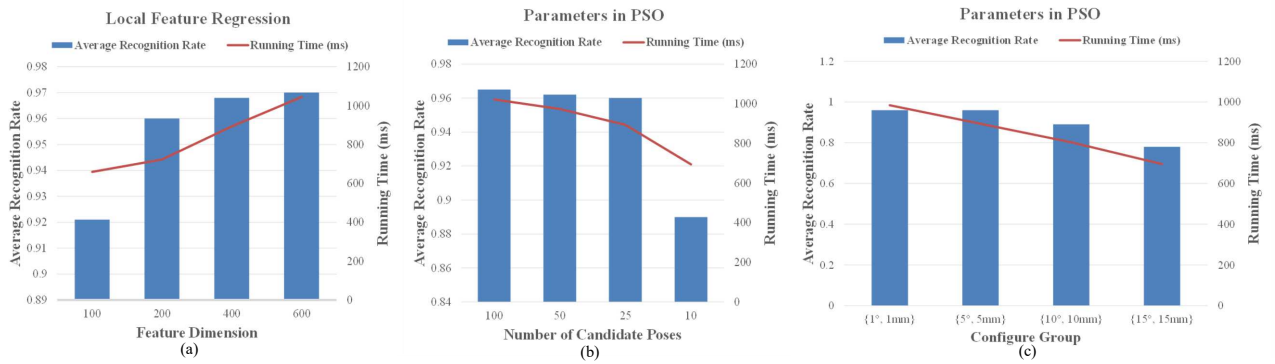


Figure 9. Evaluation of parameter influence on the LINEMOD dataset. (a) The dimension of local patch feature, (b) number of candidate poses N in PSO, and (c) configure group $\{\theta, \tau\}$ in PSO.

4.3. Discussion

There are several parameters in our method. As for local patch regression, the dimension of local patch feature is very important. Different feature dimensions will lead to different performance. In experiments, we varied the dimension of local patch feature to be 100, 200, 400, and 600. The change in average recognition rate of LINEMOD dataset [5] and running time is shown in Figure 9. It is found that larger dimension of feature has higher average recognition rate, but the overall running time increases drastically. In practical, we use the CAE with 200 feature vectors. It has enough recognition rate with acceptable running time.

In addition, the impact of PSO parameters needs to be considered, including the number of candidate poses, the range of rotation angle θ and the range of displacement τ. As shown in Figure 9, a set of decreasing number of candidate poses is used to examine the performance of our method. It is found that the number of candidate poses represents a tradeoff between accuracy and running time. A small number will lead to rapid convergence and low accuracy in fine 6D pose estimation, while a large one will prolong running time without any not noticeable improvements in accuracy. In practical, the number of candidate poses is fixed as 25 in our method. Given a fixed N, the range of rotation angle and displacement for each iteration in PSO are also need to be taken into account. In experiments, we set up some different configuration groups as shown in Figure 9. A large range will lead to less running time but low accuracy, while too small range won't have any improvement on the accuracy. From those figures, the number of candidate poses 25 and configure group {5°, 5 mm} undoubtedly constitute a good tradeoff between average recognition rate and running time of our method. And the average running time is around 800 ms per frame. The implementation of our method is based on a modern laptop PC with Inter Core i7 CPU, 8 GB memory and NVIDIA GTX 970m graphics card.

Moreover, we don't employ ICP algorithm to the stage of fine 6D pose estimation. It is well-known that ICP is sensitive to initialization, converging correctly only when sufficient overlap exists between two point sets and no gross outliers are present. In practical, the scene point cloud is always incomplete, and model point cloud is complete. If we employ ICP to refine 6D pose, we need segment scene point cloud based on rough 6D pose at first, and we need match incomplete scene point cloud with complete model point cloud. ICP often gets stuck in local minima, especially the target object has lower geometry complexity or scene point cloud has less points. In practical, incomplete scene point cloud will lead to much more error point correspondences. We found that Zabulis et al. [34] compared the performance of ICP with PSO, and it showed that PSO could get smaller 6D pose error than ICP. Because the cost function of PSO is only related with depth image and rendered depth image, the result won't be effected by incomplete scene point cloud.

5. Conclusion

Our method combines holistic patches and local patches together to fulfil the task of recovering 6D object pose. And it also achieves high precision and good performance under various conditions including foreground occlusion and background clutter, even on the challenging Occlusion dataset.

Acknowledgment

This work was funded by National Natural Science Foundation of China (Grant No. 61673261) and YASKAWA Electric Corporation.

References

[1] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. IJCV, 20(2), 2004.

[2] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose Tracking from Natural Features on Mobile Phones. In ISMAR, 2008.

[3] M. Martinez Torres, A. Collet Romea, and S. Srinivasa. Moped: A scalable and low latency object recognition and pose estimation system. In ICRA, 2010.

[4] J. Tang, S. Miller, A. Singh, and P. Abbeel. A textured object recognition pipeline for color and depth image data. In ICRA, 2012.

[5] 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 ICCV, 2011.

[6] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In ACCV, 2012.

[7] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. Gradient Response Maps for Real-Time Detection of Textureless Objects. PAMI, 2012.

[8] R. Rios-Cabrera and T. Tuytelaars, Discriminatively trained templates for 3d object detection: A real time scalable approach. In ICCV, 2013.

[9] W. Kehl, F. Tombari, N. Navab, S. Ilic, and V. Lepetit. Hashmod: A Hashing Method for Scalable 3D Object Detection. In BMVC, 2015.

[10] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6D Object Pose Estimation using 3D Object Coordinates. In ECCV, 2014.

[11] E. Brachmann, F. Michel, A. Krull, M. Yang, S. Gumhold, and C. Rother. Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In CVPR, 2016.

[12] A. Tejani, D. Tang, R. Kouskouridas, and T. Kim. Latent-Class Hough Forests for 3D Object Detection and Pose Estimation. In ECCV, 2014.

[13] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning Rich Features from RGB-D Images for Object Detection and Segmentation. In ECCV, 2014.

[14] S. Gupta, P. Arbelaez, R. Girshick, and J. Malik. Aligning 3D Models to RGB-D Images of Cluttered Scenes. In CVPR, 2015.

[15] P. Wohlhart and V. Lepetic. Learning Descriptors for Object Recognition and 3D Pose Estimation. In CVPR, 2015.

[16] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T. Kim. Recovering 6D Object Pose and Predicting Next-Best-View in the Crowd. In CVPR, 2016.

[17] 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 ECCV, 2016.

[18] Y. Jia, E. Shelhamer, J.Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. Tech. rep. (2014), http://arxiv.org/abs/1408.5093

[19] H. Cai, T. Werner, and J. Matas. Fast Detection of Multiple Textureless 3-D Objects. In ICCV, 2013.

[20] T. Hodan, X. Zabulis, M. Lourakis, S. Obdrzalek, and J. Matas. Detection and Fine 3D Pose Estimation of Texture-less Objects in RGB-D Images. In IROS, 2015.

[21] H. Zhang and Q. Cao. Texture-less object detection and 6D pose estimation in RGB-D images. Robotics and Autonomous Systems, 95 (2017): 64–79.

[22] D. Damen, P. Bunnun, A. Calway, and W. Mayol-Cuevas. Real-time Learning and Detection of 3D Texture-less Objects: A Scalable Approach. In BMVC, 2012.

[23] R. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (FPFH) for 3D registration. In ICRA, 2009.

[24] R. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In IROS, 2010.

[25] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3D object recognition. In CVPR, 2010.

[26] C. Choi and H. Christensen. 3D pose estimation of daily objects using an RGB-D camera. In IROS, 2012.

[27] M. Sun, B. Xu, G. Bradski, and S. Savarese. Depth-encoded hough voting for joint object detection and shape recovery. In ECCV, 2010.

[28] J. Gall, A. Yao, N. Razavi, L. Gool, and V. Lempitsky. Hough forests for object detection, tracking, and action recognition. IEEE transactions on pattern analysis and machine intelligence 33.11 (2011): 2188-2202.

[29] A. Krull, E. Brachmann, F. Michel, M. Yang, S. Gumhold, and C. Rother. Learning analysis-by-synthesis for 6D pose estimation in RGB-D images. In ICCV, 2015.

[30] A. Crivellaro, M. Rad, Y. Verdie. K. Yi, P. Fua, and V. Lepetit. A novel representation of parts for accurate 3D object detection and tracking in monocular images. In ICCV, 2015.

[31] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014.

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In CVPR, 2015.

[33] M. Everingham, L.V. Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. International journal of computer vision 88.2 (2010): 303-338.

[34] X. Zabulis, M. Lourakis, and P. Koutlemanis. 3D Object Pose Refinement in Range Images. In ICVS, 2015.

[35] J. Shotton, B. Glocker, C. Zach, S. Izadi, A.Criminisi, and A. Fitzgibbon. Scene Coordinate Regression Forests for Camera Delocalization in RGB-D Images, In CVPR, 2013.

[36] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige. Going Further with Point Pair Features. In ECCV, 2016.

[37] F. Michel, A. Kirillov, E. Brachmann, A. Krull, S. Gumhold, B. Savchynskyy, and C. Rother. Global Hypothesis Generation for 6D Object Pose Estimation. In CVPR, 2017.