2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Efficient Pose Estimation from Single RGB-D Image via Hough Forest with Auto-context

Huixu Dong, Dilip K.Prasad, Qilong Yuan, Jiadong Zhou, Ehsan Asadi, I-Ming Chen, *Fellow, IEEE*

***Abstract*—We propose a high efficient learning approach to estimating 6D (Degree of Freedom) pose of the textured or texture-less objects for grasping purposes in a cluttered environment where the objects might be partially occluded. The method comprises three main steps. Given a single RGB-D image, we first deploy appropriate features and the random forest to deduce the object class probability and cast votes for the 6D pose in Hough space by joint regression and classification framework, adopting reservoir sampling and summarizing the pose distribution by clustering. Next, we integrate the auto-context into cascaded Hough forests to improve the efficiency of learning. Extensive experiments on various public datasets and robotic grasps indicate that our method presents some improvements over the state-of-art and reveals the capability for estimating poses in practical applications efficiently.**

I. Introduction and Related Work

As robotic systems are getting employed in unstructured environments, they are required to manipulate objects in highly cluttered scenes[3]. A typical scenario involves pick-and-place tasks, where a robot has to estimate the pose of an object in a cluttered environment, pick up the object, and move it to a designated position, as shown in Fig. 1. A large number of efforts are invested in tackling the pose estimation problem. Although the associated challenges have been partly handled in previous works, occlusions among multiple objects can make pose estimation very difficult from a single viewpoint, supporting several ambiguous pose hypotheses. Moreover, even with an appropriate image descriptor, few of the approaches can run at very high speed because of large collection of relevance vectors14.

In this work, we focus on a specific scenario where the input is a single RGB-D image. Our target is to design a fast algorithm to generate a high-quality shortlist of candidates for 6D (3D rotation and 3D translation) poses of everyday object space to a 3D coordinate in camera space even in the objects (both textured and texture-less) accurately, which

Huixu Dong, Dilip K.Prasad, Qilong Yuan, Jiadong Zhou, Ehsan Asidi, I-Ming Chen are with Nanyang Technological University, 639798 Singapore (e-mail:dong0076@e.ntu.edu.sg).

Fig. 1. Pose estimation of an object in a cluttered environment. The result of pose estimation is visualized by the green bounding box.

transforms a 3D coordinate in the target presence of occlusions. We employ Hough forest coupled with auto-context to estimate 6D pose reliably despite the possible occlusion in a cluttered environment by means of reservoir sampling and voting regression distribution. The proposed method benefits from this integration and achieves results that are superior, in terms of detection speed and accuracy, to the recent works.

Random forests [7, 8] have been applied in several computer vision applications [10, 11] because of their ability to deal with large training datasets and fast computation [13]. Recent works make use of random forests in mapping image features to vote in a generalized Hough space [4, 13-15]. Hough voting with small variance is used for predicting object pose, where small variances indicate high detection accuracy [2, 10, 16]. However, the Hough voting step results in considerable computational effort. A random Forest trained for pose estimation will generate a large number of votes to be processed, limiting the approach's application. Recently, several methods[18, 19] of human detection and estimating human joint pose are proposed to improve the voting efficiency. Several algorithms [20-22], which are directly related to auto-context, used contextual beliefs as a weak learner in the boosting algorithm. The auto-context algorithm integrates rich image appearance models together with the context information by learning a series of classifiers, introduced by [23]. It directly targets the posterior through iterative steps, resulting in a simpler and more efficient algorithm.

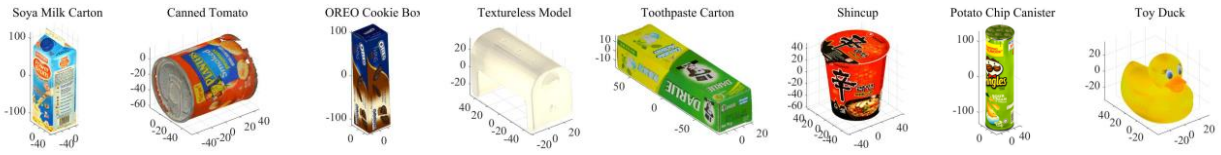

Fig. 2. 3D models of the eight textured and texture-less objects.

II. METHODOLOGY

A. Construction of Hough Forest

Decision trees in a Hough forest are constructed according to a standard random forest framework [7]. After building 3D models of the eight objects to be grasped, we render training images using a virtual camera[24](see Fig. 2). A patch extracted with the size of V from the training image has four channels (RGBD) quantized by a vector of size $V \times V \times 4$. The data thus extracted is provided as input to Hough forest.

1) Training

The appearance of the i^{th} local patch $\mathcal{P}_i$ where $\mathcal{P}_i$ is a 3D patch (e.g. of $V \times V \times 4$) sampled from RGB-D image is composed of several components: $\{\mathcal{P}_i = (\mathcal{I}_i, c_i, \theta_i, s_i, d_i)\}$. $\mathcal{I}_i$ are extracted multiple features at a patch, $\mathcal{I}_i = (I_i^0, I_i^1, I_i^2, \cdots, I_i^F)$ is a feature channel at the patch i and F is the total number of feature channels. In our system, the feature channels, such as depth [18], colour in LAB space, the first and second order gradients in x and y dimensions for the intensity space, LBP, and HoG, are used. c_i is the object class. The vector $\theta_i = \{\theta^x, \theta^y, \theta^z, \theta^{yaw}, \theta^{pitch}, \theta^{roll}\}$ contains the pose parameters associated to each patch. The components θ^x, θ^y, and θ^z represent position parameters from the point in the RGB-D camera falling on the centre of the training patch to the object position, while $\theta^{yaw}, \theta^{pitch}$ and θ^{roll} are the object rotation angles denoting the object orientation. s_i is a binary class label (0 for background sample and 1 for object patch) and d_i is the offset from the centroid of the bounding box to the centre of the patch.

Each non-leaf node B of a tree is assigned a binary test in relation to the patch appearance during training. The training sample is passed to the left or right node depending on the following binary test. The binary test $V_{B,R_1,R_2,\tau}(\mathcal{I})$ is defined as

$$V_{B,R_1,R_2,\tau}(\mathcal{I}) = \begin{cases} 0(left), \frac{1}{R_1}\sum_{i\in R_1} \mathcal{I}_i^g - \frac{1}{R_2}\sum_{i\in R_2} \mathcal{I}_i^g < \tau \\ 1(right), \quad otherwise\ , \end{cases} \quad (1)$$

where $\mathcal{I}_i^g$ represents the feature channel, R_1 and R_2 denote two rectangles around the pixel i, and τ is a threshold. The difference between the average values of two rectangular areas rather than single pixel differences is used in testing, which is less sensitive to noise. At each node during training, a pool of binary tests is generated with random values of R_1, R_2, g.

Depending on the classification to determine the object class label, the ideal binary test is the one with the minimal entropy stored in a patch in all the tests at each node as follows,

$$U_1(A) = -|A| \cdot \sum P_c \, ln(P_c) \quad (2)$$

where $|A|$ is the number of patches in a set A and P_c is the proportion of patches with the object class label c in a set A.

Traversing the binary tree costs much computational resource for training samples while training. To keep memory consumption reasonable, the training samples that arrive at the leaf nodes are summarised by the reservoir sampling [25] to obtain a fixed-size unbiased training samples. The effect of the reservoir capacity on estimation accuracy of object pose is discussed later. The mean shift is applied to clustering training samples after the reservoir sampling. The detail algorithm is shown in Table I. According to the regression, we model the object 6D pose θ at each node as realizations of a random variable with a multivariate Gaussian distribution[11], *i.e.*, $p(\theta) = \mathcal{N}(\theta; \tilde{\theta}, \Gamma)$. $\tilde{\theta}$ is the mean of 6D pose θ and Γ represents the full covariance matrix for each mode.

$$U_2(A) = \ln(|\Gamma(P)|) - \sum_{j\in\{\mathcal{L},\mathcal{R}\}} \omega_j \ln(|\Gamma_j(P_i)|), \quad (3)$$

where $\mathcal{L}$ and $\mathcal{R}$ represent the left and the right, respectively; P_i is the set of patches reaching node j and P is the set of patches at the parent node of j; ω_j is the ratio between the number of patches in the node j and in its parent node, *i.e.*, $\omega_j = \frac{P_i}{P}$. Maximizing Eq.(3) encourages minimizing the determinant of the covariance matrix Γ, thus decreasing the uncertainty for casting the votes by each patch cluster. The covariance matrix $\Gamma = \text{diag}(\Gamma^l, \Gamma^a)$ is block-diagonal, Γ^l and Γ^a denote the covariance matrix among the location vectors and among the rotation angle vectors, respectively. Thus, we can obtain the following equation,

$$U_2(A) = \ln(|\Gamma^l| + |\Gamma^a|) - \sum_{j\in\{\mathcal{L},\mathcal{R}\}} \omega_j \ln(|\Gamma_j^l| + |\Gamma_j^a|). \quad (4)$$

2) Testing

Given a novel RGB-D image, we first extract sample patches from the image which have been pre-processed using mean normalization for speeding up the processing. For each node, the stored binary test can determine the patch sample to the left or right child until a leaf.

Note that the object class is first determined independently in Hough space where we accumulate the votes of the leaf nodes. The votes are weighted according to the probability of the associated object class stored in the leaf. For the trained trees in a random forest $\mathcal{F}$, a patch $\mathcal{P}_i$ arrived in the

leaf nodes, the probability $p(c|\mathcal{F},\mathcal{P}_i)$ of object class c is calculated by averaging the probabilities of object class labels at the reached leaf nodes $L_n(\mathcal{P}_i)$ that have been stored during the period of training as follows,

$$p(c|\mathcal{F},\mathcal{P}_i) = \frac{1}{N}\sum_{n=1}^{N} p(c|L_n(\mathcal{P}_i)), \quad (5)$$

$$p(c|L_n(\mathcal{P}_i)) = \frac{1}{m}\sum_{o=1}^{m} P_c^{L(o)}(n), \quad (6)$$

where $p(c|L_n(\mathcal{P}_i))$ is defined as the probability of object class c in the reached leaf nodes of the trained tree n in $\mathcal{F}$ including N trees. $P_c^{L(o)}(n)$ denotes the proportion of patches belonging to an object class label c arriving at the leaf o of the tree n and m represents the total number of leaf nodes recording the patch $\mathcal{P}_i$ in the tree n.

At each leaf node l, a sample patch provides an estimate for the pose in terms of the stored distribution $p(\theta|l) = \mathcal{N}(\theta;\tilde{\theta},\Gamma)$,i.e., a continuous regression output. The variables are summed to produce a Gaussian; its mean is the ultimate estimate of the output parameters and its covariance evaluates the estimate's uncertainty based on mean shift interactions.

To speed up, we incorporate a clustering pre-processing step to reduce votes per node. The modes of votes are found by mean shift and sizes of distributions are employed as weights while accumulating votes. We represent the distribution using a set of the cluster centres of the K largest modes found by mean shift. We assign a confidence weight to each chosen cluster according to the size of its cluster, whereby the set of absolute votes cast by all the training samples for the object pose is aggregated using mean shift. Our experiments later highlight the importance of this. We first locate the object position and then, to estimate orientation parameters. Specifically, the mean of all votes returned by the forest is used for the initialization of the mean-shift, which is regarded as the true object position, where most of the votes usually cluster. Such cluster centres are voted in Hough space. The next step is to extract maxima in the smoothed Hough space using non-maximum suppression (NMS). The voting procedure is as shown in Fig. 3.

B. *Stacked Hough Forest coupled by Auto-context*

The auto-context is integrated into the Hough forest algorithm while learning. We train a stack of Hough forests $\{\mathcal{F}^1,\ldots,\mathcal{F}^\gamma,\ldots,\mathcal{F}^a\}$ in the cascaded way, where a represents the total number of stack forests. The whole output of the forest is taken as input to the next forest such as object class label probability $p(c|\mathcal{F}^\gamma,\mathcal{P}_i)$ and estimated 6D pose prediction $p(\theta|\mathcal{F}^\gamma,\mathcal{P}_i)$ in the γth Hough forest.

Outliers have a critical influence on accumulators in Hough space, leading in inaccurate results. To relieve this issue, we also have to smooth the predictions from the forest $\mathcal{F}^\gamma$ for the outputs of neighbouring pixels before passing to the next forest $\mathcal{F}^{\gamma+1}$. It is well-known that L1-loss function is more robust and is generally not affected by outliers. However, the instability property of L1-loss means that, for a small change of input data, the accumulators may fluctuate significantly. In contrast, L2-loss is stable. Therefore, we combine L1-loss and L2-loss functions together by means of linear functions to discard undesired outliers and improve clustering stability during iterations as much as possible. A median filter is used in a local neighbourhood of each pixel.

TABLE I. The detail algorithm of regression models

Algorithm: Regression models to summarize the distributions of training samples at leaf nodes

```
// Collect training samples arriving at leaf nodes
For all training samples i do
    Calculate the binary test
    Determine training sample to the left or right
    Descend tree to reach leaf node l
    Store training samples with reservoir sampling
// Cluster
For all leaf nodes do
    For all the training samples do
        Cluster the training samples using mean shift
        Store the mean of elements in pose vector
```

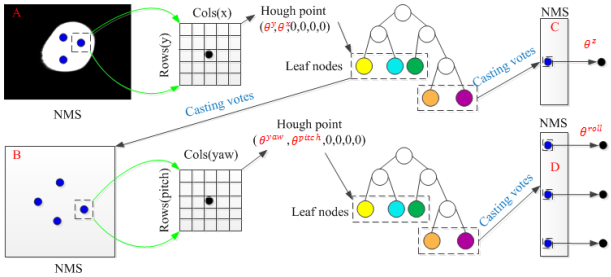

Fig. 3. Detection for object class and pose by casting votes. A,B,C,D represent the voting spaces for object class and 6D pose, respectively.

Under L1-loss and L2-loss, the median-smoothed object probabilities $p(c|\mathcal{F}^\gamma,\mathcal{P}_i)$ are defined as

$$h_c^\gamma(c|\mathcal{P}_i,L_1) = argmin\ \sum_{i\in\mathcal{N}_i}\|p(c|\mathcal{F}^\gamma,\mathcal{P}_i) - \tilde{p}_i\|_1, \quad (7)$$

$$h_c^\gamma(c|\mathcal{P}_i,L_2) = argmin\ \sum_{i\in\mathcal{N}_i}\|p(c|\mathcal{F}^\gamma,\mathcal{P}_i) - \tilde{p}_i\|_2, \quad (8)$$

where we denote by $\mathcal{N}_i$ a small neighbourhood around the centre pixel i in $\mathcal{P}_i$. $\tilde{p}_i$ is the median object probability for each local neighborhood.

Similarly, we also use the mode above to smooth the 6D pose prediction $p(\theta|\mathcal{F}^\gamma,\mathcal{P}_i)$,which is crucial for predicting object 6D pose. The median filter is not only robust to outliers but also stable for clustering based on L1-loss and L2-loss. Since the median filter is just used in smoothing data with one dimensionality. the optimum under L1-loss has to be formulated in Euclidean space. Thus, the regularized object 6D pose output is defined as follows,

$$h_\theta^\gamma(\theta|\mathcal{P}_i,L_1) = argmin\sum_{i\in\mathcal{N}_i}\sum_{T\in\mathcal{F}^\gamma}\left\|\mu_{i,T} - \theta\right\|_2, \quad (9)$$

$$h_\theta^\gamma(\theta|\mathcal{P}_i,L_2) = argmin\sum_{i\in\mathcal{N}_i}\sum_{T\in\mathcal{F}^\gamma}\left\|\left(\mu_{i,T} - \theta\right)^2\right\|_2, \quad (10)$$

where $\mu_{i,T}$ represents the mean value with the highest mixture weight of the pose distribution $p(\theta|\mathcal{F}^\gamma,c,\mathcal{P}_i)$ for the object class c in the tree T from the forest $\mathcal{F}^\gamma$. To discard undesired outliers and improve clustering stability during

iterations as much as possible, we combine L1-loss and L2-loss functions together by means of linear functions. The final median-smoothed probability is provided as

$$h_c^{\gamma}(c|\mathcal{P}_i) = \lambda_1 h_c^{\gamma}(c|\mathcal{P}_i, L_1) + \lambda_2 h_c^{\gamma}(c|\mathcal{P}_i, L_2), \quad (11)$$

$$h_\theta^{\gamma}(\theta|\mathcal{P}_i) = \lambda_1 h_\theta^{\gamma}(\theta|\mathcal{P}_i, L_1) + \lambda_2 h_\theta^{\gamma}(\theta|\mathcal{P}_i, L_2), \quad (12)$$

where we denote by λ_1 and λ_2 the weights of $h_c^{\gamma}(c|\mathcal{P}_i, L_1)$, $h_\theta^{\gamma}(\theta|\mathcal{P}_i, L_1)$ and $h_c^{\gamma}(c|\mathcal{P}_i, L_2)$, $h_\theta^{\gamma}(\theta|\mathcal{P}_i, L_2)$,respectively. We set $\lambda_1 = \lambda_2 = 0.5$. This output for this feature $f_c(x) = h_c^{\gamma}(c|\mathcal{P}_i, u + o)$ on the object class label, is used in forest $\mathcal{F}^{\gamma+1}$. The location u at the pixel i and the relative offset o,the smooth object class c are included the parameter x. Correspondingly, we define the feature $f_\theta(x) = h_\theta^{\gamma}(\theta|\mathcal{P}_i, u + o)$ on the smooth object pose output. The parameter x consists of the object class c, the pose θ, the relative offset o and the location u at the pixel i. The set of passed features can be enhanced by updating $f_c(x)$ and $f_\theta(x)$ at the current forest.

C. *Refining the Final Pose*

Through the verifications above, an object has multiple pose candidates because of a large number of votes. Here, we use the method in [4] to refine the object pose.

III. Experiments and Discussions

Here we explain three general evaluation metrics. For Metric 1 defined in [1], the score that evaluates the difference between the ground truth pose and the estimated pose is used in determining whether the estimated pose is correct or not. If the translational error and angular error is less than 5cm and 5° respectively, the estimated pose is considered correct, which is named by Metric 2[26]. We also use F-measure as the criterion, which is the harmonic mean of precision and recall.

A. *Framework Parameters*

Parameter optimization is performed on a validation dataset created by a virtual camera. For each object class, we train Hough Forest with varying parameters.

The size V of the patch has an important effect on the detection accuracy because a large patch tends to generate a holistic matching, which is sensitive to clutters and occlusions. However, if the size is very small, it may be regarded as noise. Note that the decrease of F-measure occurs for patches with size equal to 20 and higher because of clutters or occlusions (see Fig. 5(A)). Evidently, a forest with more than four trees imposes the computational burden. In this work, we set the number of trees to 4 and the size V of the patch is 20.

Figure 5(B) shows that F-measure rapidly improves as the tree depth increases, though it starts to level off around depth 24. For 50k images, the over fitting occurs at about depth 20, but it is avoided using the enlarged 100k training images. The deeper tree obtains a large extra memory penalty. Here the maximum depth of trees is set 26 and the number of training images is 100k.

The effect of the number of forests on the detection performance is shown in Fig. 5(C), where it is obvious that the detection accuracy increases gently when the number of Hough forests is larger than 3 while the execution time consistently increases. Thus, the best number of random forest is set as 3.

With increasing the number of stride, F-measure decreases in Fig. 5(D). Thus, we set the stride 4.

From Fig. 5(E), the size of the reservoir has little effect on the detection accuracy. From 200 to 300, a small increase occurs in F-measure. In surprise, increasing the reservoir capacity to 600 leads to a small drop. To quantify the role of the number K of cluster centers, we test our system varying the number of cluster centers. Increasing the number of cluster centers causes a small rise decreases from 0.889 to 0.902, as shown in Fig. 5(F). We set this number of cluster centers 100.

Figure 5(G) shows the effect with and without reservoir sampling and the distributions to represent votes by clustering on the execution time. It reveals that the use of these two methods can improve the detection efficiency significantly.

B. *Performance Evaluation by Self-comparison*

As shown in Fig. 5(H), the integration of the auto context into Hough forest evidently improves the performance of 6D pose estimation. Also, the combination of L1-loss and L2-loss boosts the detection performance. Without auto-context, the presented estimator fails to estimate the correct poses reliably. For the metric in [1], our approach is very close to 100% . In contrast, since the metric in [26] is more rigorous than the metric in [1] for rotation evaluation, the presented method still achieves the accuracy of 83.6%. Using the combination of L1-loss and L2-loss, compared to L1-loss only or just L2-loss only, results in improvements of 8.3% or 10.9% for the metric[1] and 4.1% or 6.5% for the metric[26], respectively.

C. *Performance Evaluation of Two Public Datasets*

To illustrate that our approach can address the pose estimation of multiple objects in an image, we evaluate our method on the dataset presented by Tejani et al.[2]. The comparison results of the several methods regarding F-measure are presented in Table II. The challenge of estimating poses of “Camera” and “Milk” instances stems from the main factor that these objects are texture-less and of uniform colour in cluttered environments. Our approach is superior to the state of arts on the Camera and Milk box that has the similar appearance with the clutter and is partially occluded. This also shows the robustness of our method to the cluttered scenarios. We obtain an average F-measure of 0.907 in the dataset of Tejani, which excesses 2.2% than the current state-of-the-art method[6]. Several examples are shown in Fig. 4.

The occlusion dataset created in [5] is an extension of [1], which includes instances of multiple objects occurring with

serious occlusions. In this case, it is difficult to get the pose of an object due to the partial occlusions. However, our method still can achieve an average rate of correctly estimated poses of 81.3% using the criteria presented in [1]. The qualitative examples are illustrated in Fig. 4. Here, our method provides excellent accuracy for estimating the pose of "Ape" that has a small size, which performs best among these methods. In comparison with the newest work implemented in [17], our approaches presents improvements of 4.6%(Table III).

D. Robotic Grasp Experiments

The setup is shown in Fig. 1. All the experiments are performed on a PC with 15.6 GiB Memory, GTX 645 and an Intel Core i7 processor. In the first set of grasp experiments, our goal is to grasp 6 objects that are placed at 5 different poses (i.e., random position and orientation) using a robot. A successful grasp is defined that the robotic gripper lifts an object 15cm above the supporting plane and keeps it around 5 seconds. We first place an object at a random pose on a table in front of the robot. Depending on the estimated pose, the robot plans a suitable path to be close to the object in a "pre-grasp" position, which is defined as 10 cm away from the actual grasping configuration along the horizontal direction. Then the gripper moves to the actual grasp position, grasps the object, and lifts it to a position 15 cm higher. Figure 6(the first row) shows the grasp steps. Several grasp cases are shown in Fig. 6(the second row). Moreover, the grasp success rates for each object are also summarized in Table IV. The robot successfully grasped 28 objects out of 30 tries, realizing an overall success rate of 93%. Successful grasps indicate our approach to estimating 6D pose can be used in a real world. We find that our algorithm of pose estimation is quite robust to the objects with multiple poses in real scenarios. In general, two failures were due to incomplete capture of depth information, which resulted in the fingertip colliding with the grasped object. The average estimation time of the proposed method can achieve 0.92s per RGB-D image.

IV. CONCLUSION

We proposed a new approach of estimating the 6D poses of objects in crowed scenes by means of Hough Forests coupled with auto-context from raw RGB-D data. Our method consistently outperforms state-of-the-art techniques on standard benchmark datasets and significantly improves robotic capabilities in targeted object manipulation. As a future work, we want to investigate how to optimize this algorithm so that a robot can grasp an object in real time.

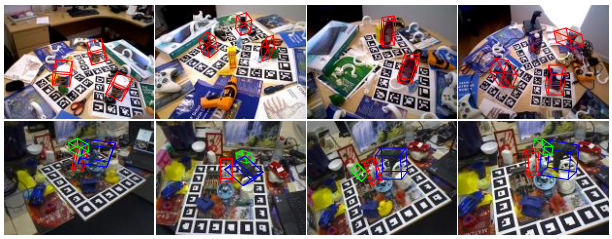

Fig. 4. The examples of pose estimation on the dataset of Tejani(the first row) and the occlusion dataset (the second row).

TABLE II. Results of F-measure.

Object	Method						
	L.[1]	T.[2]	D.[4]	B.[5]	K.[6]	Ko.[9]	Ours
Camera	0.589	0.394	0.903	0.691	0.741	0.784	**0.905**
Coffee	0.942	0.891	0.932	0.912	**0.983**	0.946	0.979
Joystick	0.846	0.549	0.924	0.759	**0.997**	0.741	0.956
Juice	0.595	0.883	0.819	0.897	0.919	**0.935**	0.906
Milk	0.558	0.397	0.510	0.476	0.780	0.516	**0.799**
Shampoo	**0.922**	0.792	0.735	0.824	0.892	0.882	0.890
Average	0.740	0.651	0.803	0.759	0.885	0.800	**0.907**

TABLE III. Results of the accuracy.

Object	Method				
	L.[1]	K.[12]	B.[5]	M.[17]	Ours
Ape	81.4%	68.0%	53.1%	80.7%	**87.3%**
Can	**94.7%**	87.9%	79.9%	88.5%	89.5%
Cat	55.2%	50.6%	28.2%	57.8%	**62.3%**
Driller	86.0%	91.2%	82.0%	94.7%	**98.5%**
Duck	79.7%	64.7%	64.3%	74.4%	**79.8%**
Eggbox	65.5%	41.5%	90.0%	47.6%	**68.2%**
Glue	52.1%	65.3%	44.5%	**73.8%**	66.8%
Puncher	95.5%	92.9%	91.6%	96.3%	**97.0%**
Average	76.3%	70.3%	56.6%	76.7%	**81.3%**

TABLE IV. Experimental results of robotic grasps

Object	Trials	Pre-grasp correct	Success times	Success rate	Average time(s)
Milk box	5	100%	4	80%	0.89
Peanut can	5	100%	5	100%	0.71
Duck toy	5	100%	5	100%	0.75
Tooth-paste box	5	100%	5	100%	0.37
White part	5	100%	5	100%	1.23
Potato chip box	5	80%	4	80%	0.63
Results	30/t.	96.67%/a.	28/t	93%/a.	0.92/a.

t. and a. mean the total value and the average value, respectively.

REFERENCES

[1] S. Hinterstoisser et al., "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in Asian conference on computer vision, 2012, pp. 548-562: Springer.

[2] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim, "Latent-class hough forests for 3D object detection and pose estimation," in European Conference on Computer Vision, 2014, pp. 462-477: Springer.

[3] H. Dong, G. Sun, W.-C. Pang, E. Asadi, D. K. Prasad, and I.-M. Chen, "Fast Ellipse Detection via Gradient Information for Robotic Manipulation of Cylindrical Objects," IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 2754-2761, 2018.

[4] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T.-K. Kim, "Recovering 6D object pose and predicting next-best-view in the crowd," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3583-3592.

[5] 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, 2014, pp. 536-551: Springer.

[6] 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 Conference on Computer Vision and Pattern Recognition, 2017, pp. 1521-1529.

[7] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5-32, 2001.

[8] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera

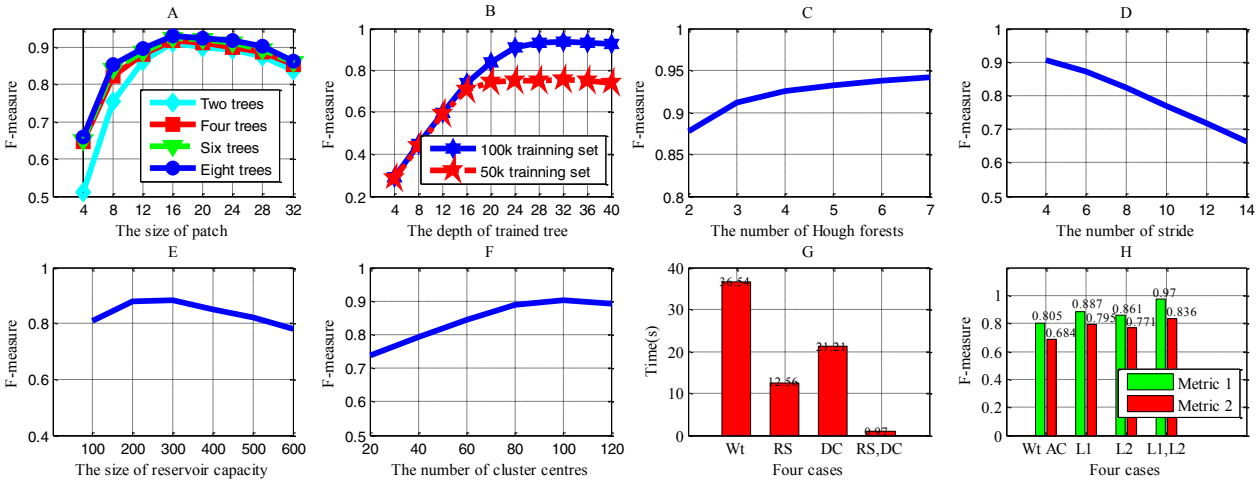

Fig. 5. Parameter determination and self-comparison. Wt(Without auto-context),RS(Just reservoir sampling),DC(the distribution by clustering). Wt AC(Without auto-context),L1(L1-loss),L2(L2-loss),(Metric 1) and (Metric 2).

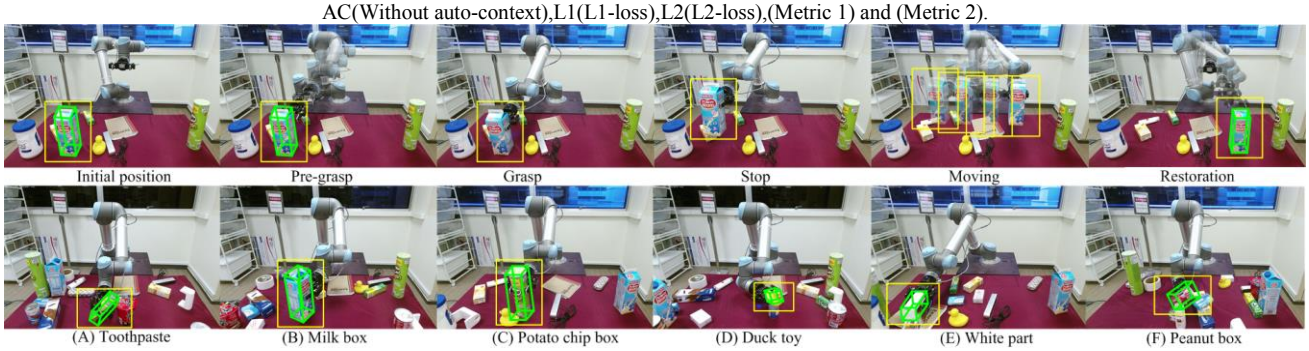

Fig. 6. The steps of robot grasping an object (the first row) and robot grasping several different textured and texture-less objects in cluttered environments.

relocalization in RGB-D images," in Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, 2013, pp. 2930-2937: IEEE.

[9] R. Kouskouridas, A. Tejani, A. Doumanoglou, D. Tang, and T.-K. Kim, "Latent-class hough forests for 6 dof object pose estimation," arXiv preprint arXiv:1602.01464, 2016.

[10] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky, "Hough forests for object detection, tracking, and action recognition," IEEE transactions on pattern analysis and machine intelligence, vol. 33, no. 11, pp. 2188-2202, 2011.

[11] A. Criminisi, J. Shotton, D. Robertson, and E. Konukoglu, "Regression forests for efficient anatomy detection and localization in CT studies," in International MICCAI Workshop on Medical Computer Vision, 2010, pp. 106-117: Springer.

[12] A. Krull, E. Brachmann, F. Michel, M. Ying Yang, S. Gumhold, and C. Rother, "Learning analysis-by-synthesis for 6D pose estimation in RGB-D images," in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 954-962.

[13] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, 2011, pp. 617-624: IEEE.

[14] A. Tejani, R. Kouskouridas, A. Doumanoglou, D. Tang, and T.-K. Kim, "Latent-Class Hough Forests for 6 DoF Object Pose Estimation," IEEE transactions on pattern analysis and machine intelligence, vol. 40, no. 1, pp. 119-132, 2018.

[15] C. Redondo-Cabrera, R. López-Sastre, and T. Tuytelaars, "All together now: Simultaneous object detection and continuous pose estimation using a hough forest with probabilistic locally enhanced voting," in Proceedings BMVC 2014, 2014, pp. 1-12.

[16] J. Shotton et al., "Efficient human pose estimation from single depth images," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 12, pp. 2821-2840, 2013.

[17] F. Michel et al., "Global hypothesis generation for 6D object pose estimation," arXiv preprint, 2017.

[18] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon, "Efficient regression of general-activity human poses from depth images," in Computer Vision (ICCV), 2011 IEEE International Conference on, 2011, pp. 415-422: IEEE.

[19] K. Rematas and B. Leibe, "Efficient object detection and segmentation with a cascaded hough forest ism," in Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on, 2011, pp. 966-973: IEEE.

[20] A. Torralba, K. P. Murphy, and W. T. Freeman, "Contextual models for object detection using boosted random fields," in Advances in neural information processing systems, 2005, pp. 1401-1408.

[21] M. Fink and P. Perona, "Mutual boosting for contextual inference," in Advances in neural information processing systems, 2004, pp. 1515-1522.

[22] S. Avidan, "Spatialboost: Adding spatial reasoning to adaboost," in European Conference on Computer Vision, 2006, pp. 386-396: Springer.

[23] Z. Tu and X. Bai, "Auto-context and its application to high-level vision tasks and 3d brain image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 10, pp. 1744-1757, 2010.

[24] M. Sun, G. Bradski, B.-X. Xu, and S. Savarese, "Depth-encoded hough voting for joint object detection and shape recovery," in European Conference on Computer Vision, 2010, pp. 658-671: Springer.

[25] J. S. Vitter, "Random sampling with a reservoir," ACM Transactions on Mathematical Software (TOMS), vol. 11, no. 1, pp. 37-57, 1985.

[26] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in RGB-D images," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 2930-2937.