IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 1, 2018, accepted October 21, 2018, date of publication October 24, 2018, date of current version November 30, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2877796

A Self-Supervised Learning Manipulator Grasping Approach Based on Instance Segmentation

XIN SHU[1,2], CHANG LIU[1], (Fellow, IEEE), TONG LI[1], CHUNKAI WANG[1,2], AND CHENG CHI[1,2]

[1]National Institute of Standards and Technology State Key Laboratory of Transducer, Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China

[2]University of Chinese Academy of Sciences, Beijing 100049, China

Corresponding authors: Chang Liu (tuengineer@qq.com) and Tong Li (tli@mail.ie.ac.cn)

This work was supported in part by the Key Research Program of Frontier Science, CAS, under Grant QYZDY-SSW-JSC037 and in part by the National Natural Science Foundation of China under Grants 61774157, 61802363, and 81771388.

ABSTRACT Automatic grasping is playing an important role in robotics, and traditional grasping approaches cannot deal with both occlusions and texture-less objects well. To improve the stability and accuracy of grasping in a novel and occlusion environment, a grasping approach based on instance segmentation and self-supervised learning pose estimation network is proposed to grasp objects with the manipulator in this paper. The approach can be divided into three phases: instance segmentation, pose estimation, and pose transformation. Instance segmentation predicts classification masks for each pixel. Masks which stand for the contour of objects provide a heuristic knowledge for the self-supervised learning pose estimation network. Pose estimation network has two fully connected layers and regards estimation as a self-supervised classification problem. Then, the pose can be transformed from value in pixel coordinates to actual value relative to the base coordinate of the manipulator. As a result, the manipulator can be operated to grasp by the given actual value of pose. With the help of the approach proposed in this paper, we improve the grasping accuracy by 35%, compared to the former grasping approach based on the pose estimation network on the grasping dataset of CMU. Besides, grasping with this approach on hardware also shows a high success rate. Therefore, the proposed approach is a more robust and more accurate way of grasping.

INDEX TERMS Grasping, neural networks, robot vision systems, supervised learning.

I. INTRODUCTION

Grasping operation with manipulator is a hot research direction in recent days. The industrial manipulator grasping is based on manual teaching. A manipulator can grasp the object by fixed position after being taught to do so. It is applied in robot industry that guide robot to do some repetitive works. It is obviously that the traditional grasping scheme is hard to deal with grasping novel objects. Therefore, new grasping pose needs to be taught to make it operate normally. With the development of computer vision and Amazon Picking Challenge [1], a lot of grasping methods based on learning and template matching are becoming more and more popular. For learning-based approaches, CNNs [2], [3], and autoencoders [4] are used to predict grasping locations on the RGB image. Rodney [5], Shimoga [6], Lozano-Perez *et al.* [7], and Van-Duc [8] show the way CNNs work to predict grasping attitude with 3D depth sensor. Since the grasping system considering both location and attitude based on learning, Du *et al.* [9] proposed an approach based on deep learning which spends more than 10 seconds on estimation of grasping pose due to the large number of weights of fully-connect layers with the GPU of GTX980. However, learning-based approaches rely on texture of objects and have some disadvantages when inferring texture-less objects. There are also lots of ways based on template matching proposed. Zeng *et al.* [10] applied the network of 3D segmentation and calculates the grasping pose with template matching of 3D point cloud, which received ideal results. Some methods optimizing the mechanism of template matching such as SegICP [11] are proposed recently to improve matching accuracy. Both self-occlusion and mutual-occlusion lead to fragmentary of object information, which is harmful to template matching. Therefore, these approaches based on template matching cannot perfectly deal with self-occlusion and mutual-occlusion between objects, which are common because of the inappropriate pose of camera.

In a word, texture-less and occlusions are still big challenges for grasping pose estimation. Moreover, small objects

introduce many issues in analysis and grasping due to its low definition in vision.

Therefore, an intelligent grasping approach based on instance segmentation and self-supervision learning pose estimation, which shows robustness to multi-scale and texture-less objects and improves the grasping accuracy, is proposed in this paper. The main contribution of this paper is that the approach sends the output masks of instance segmentation to a pose estimation network which is trained by robot-labeled dataset. Feature pyramid network is set as a part of network to improve the performance of multi-scale objects. Unlike template-matching based approaches, our approach overcomes the difficulties of occlusions and can grasp objects which are not appeared in template library. Since it is trained by robot-labeled datasets, it performs well on texture-less objects in grasping, which is a great challenge for other learning-based approaches.

II. RELATED WORKS

Our approach builds on the substantial literature devoted to instance segmentation and grasping pose estimation based on networks. The whole robot system should identify the object in the task environment and have a probable location of it firstly. Then, the relative pose of the object under manipulator is needed to be estimated for manipulator operations. Therefore, literatures about instance segmentation and pose estimation are discussed here to explain how they develop in recent years.

A. INSTANCE SEGMENTATION

Compared to semantic segmentation, instance segmentation distinguishes object instances. In another word, it includes both classification and segmentation of objects. The classification task is based on region-based methods and the segmentation task is based on segment proposal methods. So, lots of approaches which process these two sub-tasks separately such as SDS [12], Hyper-column [13], CFM [14], MNC [15], Multi-PathNet [16]. To propose an end-to-end instance-aware semantic segmentation solution, Li *et al.* [17] interact the segment proposal method in [18] and object detection method in [19] to an instance segmentation system, which is called ''fully convolutional instance segmentation''. He *et al.* [20] propose the Mask R-CNN, which is regarded as a small FCN mask branch added to Faster R-CNN to predict a multi-channels output.

Approaches process tasks separately have some drawbacks such as losing spatial details because of the ROI pooling, which harms segmentation results a lot. FCIS predicts classes, boxes and masks simultaneously and it is fully convolutionally. As it is an end-to-end network, the whole inference procedure is faster than former works. Whereas it is still hard for FCIS to deal with edges of overlapping instances and the proposal of Mask R-CNN eliminates spurious edges caused by errors on overlapping instances in FCIS. We design a segmentation network for grasping, which is similar with Mask R-CNN. Occlusion is common in grasping and our network deals well on it.

B. POSE ESTIMATION

With the help of object detection and sematic segmentation, pose estimations refines object's location and calculates the most likely grasping pose. It distinguishes a lot for RGB images and RGB-D images. For RGB-D images, LINEMOD [21] uses gradient and normal features to estimate the pose. Besides, a more widely used approach is iterative closet point registration [22] which match the class of point cloud to the template. As for RGB images, there lots of local features such as SIFT [23], ORB [23] to get the pose of highly-textured objects in traditional way. To improve the robustness of estimation, several datasets and models built on deep learning are proposed such as [25] and [26].

For RGB-D images, the method based on building 3D models is an extremely huge problem by itself, which costs more calculation time. And the cost of RGB-D camera is high. As for RGB-images, the feature designed by deep learning shows better performances on estimation than traditional features, whereas these methods have disadvantages of losing stereoscopic information. We design a self-supervised pose estimation network for RGB images. As it is self-supervised, the stereoscopic information and distribution of mass are considered during the estimation.

In summary, our approach combines the instance segmentation network and the pose estimation network. This approach appears to be one of the first to take the output of segmentation as the input of following pose estimation network, which receives great performance in grasping accuracy.

III. SELF-SUPERVISED MANIPULATOR GRASPING APPROACH

In our approach, instance segmentation, pose estimation and pose transformation are mainly involved in. Procedures can be seen in FIGURE **1**. Instance segmentation consists of object detection and semantic segmentation. Object detection detects categories and the bounding box of objects in the RGB image obtained with depth camera. Semantic segmentation gets the more accurate contour information of objects and is a classifier for every pixel. The second part is the pose estimation, which is designed to calculate the best pose of the object for manipulator to grasp. It relies on the result of object segmentation and size of gripper fingertips. Finally, it is needed to transform the pose in the image to the pose under manipulator. The above three main aspects provide information for manipulator operations so that the manipulator can grasp and place objects automatically.

A. INSTANCE SEGMENTATION NETWORK FOR GRASPING

Depth camera can extract both color frames and depth frames. In our approach, we use a network which is based on Faster R-CNN [27] to analysis the classification, bounding box and contour of objects in the color frame as the instance

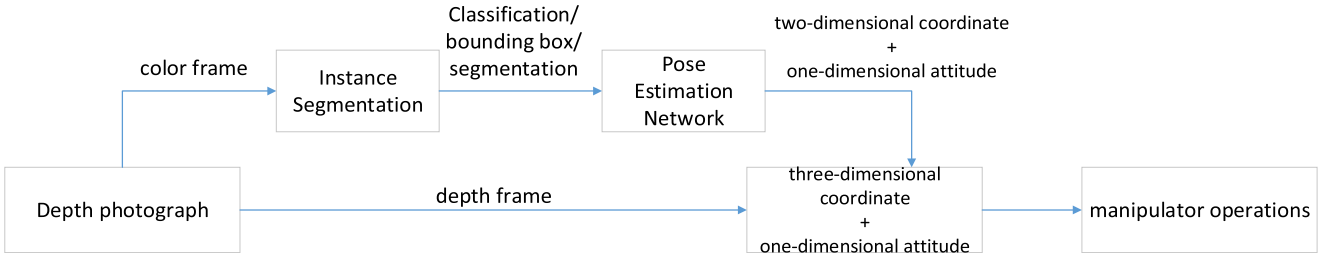


FIGURE 1. Block diagram of our grasping approach.

segmentation method. Convolutional layers are designed to extract feature map and then Faster R-CNN includes two main parts of network. The first one is called Region Proposal Network, which is designed to propose candidate object bounding boxes. And the second part uses a ROIPool layer to extract feature of proposal boxes from feature map and regress the classification and bounding box. We add a branch for predicting an object mask which can stands for the contour of object, in parallel with the existing branch for bounding box recognition on the framework of Faster R-CNN. ResNet-101 is set as the backbone of network which appears better than ResNet-50 and other kinds of art-of-state backbones. A Feature Pyramid Network [28] is added as the head of framework and uses a top-down architecture with lateral connections to build an in-network feature pyramid from a single-scale input, which improves multi-scale detection of objects. ROIAlign Layer is designed to replace the former ROIPool Layer because of its decrease in misalignments between the ROI and the extracted feature. The ROI is divided into a fixed number of bins. It removes the quantization of the ROI boundaries and bins while bilinear interpolation is applied to compute the values of input features at four regular sampled location in ROI bin. Thus, there is no more quantization to be considered.

A multi-task loss function is proposed to combine the classification loss, the bounding box loss and the mask loss while training. The classification loss and the bounding box loss are proposed in Fast R-CNN [29] and they are same with these defined in it. The mask loss has the output of KM^2 dimension which represents K classes binary masks for M∗M resolution and it is an average binary cross-entropy loss with pixel level sigmoid, which shows large gains over softmax. And it is only applied when the ROI is positive. A ratio of 1:3 of positive to negatives of sampled ROIs are sent to the GPU while training. Besides, RPN and other convolution layers share their features because they use the same backbone. When it comes to inference, 1000 ROIs are proposed per image and k^{-th} mask is outputted, where k is the predicted class of the detection branch.

The instance segmentation information is sent to the following pose estimation network. On the one hand, it can help us to grasp the exact object which we want. On the other hand, bounding box and contour of the object provided by it can improve the inference accuracy of the pose estimation network with the same quantity of sampled patches.

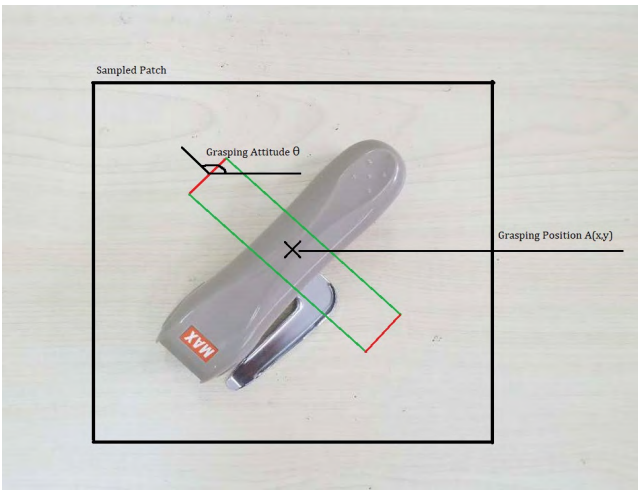


FIGURE 2. Configurations of grasp.

B. SELF-SUPERVISED LEARNING POSE ESTIMATION NETWORK

A self-supervised learning network is applied to estimate the pose of grasping. Black lines mean the sampled patch and grasping configuration lies in three parameters, grasping position A (x, y) and grasping attitude θ. As shown in FIGURE 2, grasping position is the center of grasping patch and grasping patch is set to a fixed size which is larger than the projection of gripper fingertips on the image to include context. Grasping attitude ranges from 0° to 180°, whose period is 180°. Red lines are the corresponding gripper location.

Five convolutional layers taken from AlexNet CNN model and two fully connected layers with 4096 and 1024 neurons are designed as the framework of network. When given an image, we randomly sample grasping positions and extract patches which are fed into the network to predict the grasping attitude. The grasping position is calculated as the center of sampled patch. As for attitude estimation, classification shows higher grasping rate than regression in pose estimation, which can be referred in the related works [26]. We assume that the step of grasping attitude is θ and 180 should be divided by θ. For every given patch, we estimate an $(180/\theta)$-dimensional likelihood vector which represents the likelihood of whether the center of the patch is graspable at $(180/\theta)$ different grasping attitudes. Thus, it is seen as an $(180/\theta)$-way binary classification problem.

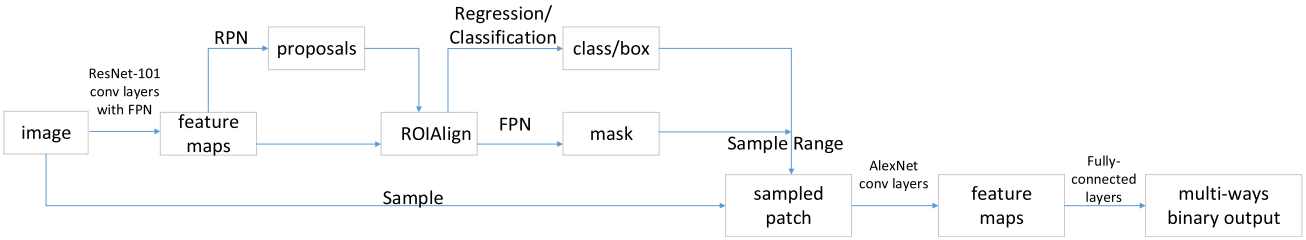


FIGURE 3. Framework of the whole network.

Softmax is added to calculate the loss function when the network is training. The dataset used in training is labeled by grasping attempts with manipulator instead of human labeling, which means it is self-supervised. Given a grasping position and a grasping attitude, the manipulator can grasp objects with the given grasping pose. After the gripper is closed and the manipulator rises, if the pressure sensor on the manipulator can feel the constant pressure during the grasping method, then the pose is labeled to positive. Otherwise, the pose is labeled to negative. The whole labeling method is based on self-supervised of manipulator. Self-supervised reduce the work of human labeling. Besides, it considers mass distribution of the object which is quite important to object grasping. While it comes to inference, we select the grasping position and grasping attitude with the highest output score from all angles and all sampled patches.

The combination of instance segmentation and pose estimation helps a lot on grasping. We sample patches inside of contour of the object. As baseline discussed in [30], there is a rule about grasp that manipulator should grasp about the center of the patch. It is implicit in the combination because the center of patch falls in the bounding box or the contour of object. FIGURE 3 shows the overall framework of our network.

C. POSE TRANSFORMATION

To make the manipulator operate normally, we need the three-dimensional coordinate and the grasping attitude under the manipulator. We can get the grasping attitude from the pose estimation network, we still need to transform the two-dimensional grasping location in the color frame to the three-dimensional grasping location under the manipulator. Theoretically, we can get depth information of the object in color frame through the calibration of color frame and depth frame to extract three-dimensional location coordinates under camera space. We take the Kinect as our depth camera and the camera space of the Kinect is shown in FIGURE 4. We use a rectangle to calibrate the color frame and depth frame. Firstly, we must let the side of rectangle parallel to the bounding of frame. Then, we measure the correspond length of a side in two frames for 10 times and calculate the average transformation ratio l_x and l_y in two bounding directions. We assume a point's pixel coordinate in the color frame is (x_c, y_c) and this point's pixel coordinate is (x_d, y_d) in the depth frame. The relationship between them can be

FIGURE 4. Camera space of Kinect.

described in the following equation.

$$\begin{aligned} x_d &= l_x * x_c + b_x \\ y_d &= l_y * y_c + b_y \end{aligned} \tag{1}$$

As the transformation ratio l_x and l_y is calculated, we sample 10 corresponding pixels in color frame and depth frame to calculate the transformation bias b_x and b_y. Similarly, we also use the average of 10 calculated values. With transformation ratio and bias, we can get the correspond pixel in depth frame for every pixel in color frame. With the help of depth information, we can calculate the three-dimensional coordinates under the camera space by following equations.

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = z_1 M_{in}^{-1} \begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix} \tag{2}$$

$$M_{in} = \begin{bmatrix} k_x & 0 & u_0 \\ 0 & k_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

In the above equation, M_{in} is the internal matrix of camera and z_1 is the depth distant obtained. So, the coordinates of grasping position under the Kinect space (x_1, y_1, z_1) can be calculated with the above equation.

However, we want to get the coordinate under the manipulator instead of space coordinate under the camera space to simplify the operation of manipulator. We adjust the relative position of Kinect and manipulator to make this question easier. As shown in FIGURE 5, we coincide the Kinect's Z axis with the arm's X axis. And we make the Kinect's X axis parallel to the arm's Y axis. In this way, the Kinect's Y axis is parallel to the arm's Z axis. We assume that coordinates of grasping position under the Kinect space is (x_1, y_1, z_1), then coordinates of the center of the object under the manipulator

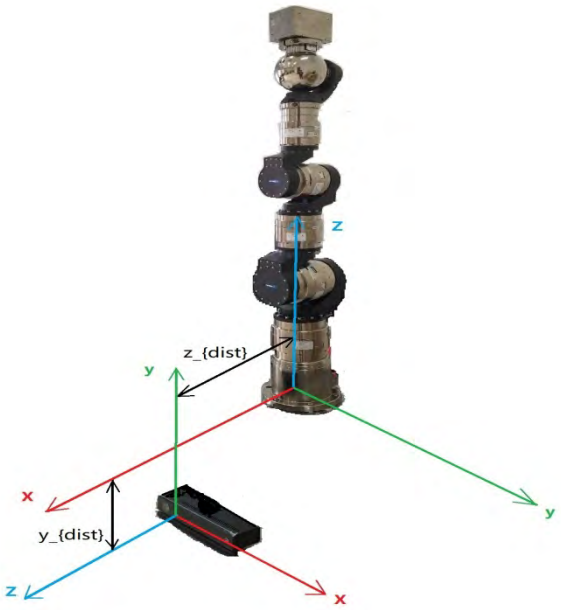


FIGURE 5. Space under Kinect and manipulator.

space is (x_2, y_2, z_2). From the following equations, we can see the relationship between them.

$$\begin{aligned} x_2 &= z\_\{dist\} + z_1 \\ y_2 &= x_1 \\ z_2 &= y_1 - y\_\{dist\} \end{aligned} \tag{4}$$

In these equations, $y\_\{dist\}$ is the distance between two coordinate origins in the Y axis of Kinect space and $z\_\{dist\}$ is the distance between two coordinate origins in the Z axis of Kinect space, which is shown in FIGURE 5. From these equations, three-dimensional space coordinates under the manipulator can be easily calculated.

Grasping operations with manipulator need three-dimensional pose information, which means six inputs totally whereas we only provide the three-dimensional position and a single attitude. The relative location relationship between the camera and the manipulator is shown in FIGURE 5. Thus, we assume that the Z axis of gripper is always perpendicular to the camera photography plane. In this condition, the manipulator can be operated with a three-dimensional position and a single attitude information. Therefore, we can grasp objects with the pose calculated in this condition. The whole operation procedure is as follows:

1) Initialize the pose of the manipulator.

2) Control the last three joints to make the Z axis perpendicular to the camera photography plane and adjust the grasping attitude.

3) Control the front three joints to reach the target position from the initialization and keep the attitude in 2).

4) Close the gripper and apply pressure to grasp the target object.

5) Control the front four joints to move the object to a fixed location and place it.

6) Turn back the manipulator to the initialization pose.

TABLE 1. Detection results on COCO.

METHOD	DATA	AP, IOU:			AP, AREA:		
		0.5:0.95	0.5	0.75	S	M	L
SSD513	VAL_2017	33.2	53.3	35.2	13.0	35.4	51.1
YOLO V3	VAL_2017	33.0	57.9	34.4	18.3	35.4	41.9
FASTER R-CNN	VAL_2017	36.2	59.1	39.0	18.2	39.0	48.2
OURS	VAL_2017	40.9	61.9	44.8	23.5	44.2	53.9

(a) (b)

FIGURE 6. (a) shows the input image where objects are overlapped and (b) is the segmentation result.

IV. EVALUATIONS

In experiments, we use Kinect as the depth camera, which can get the $1920 * 1280$ color frame and $512 * 424$ depth frame. And we train the segmentation network with the coco_train_2017 dataset. The coco dataset includes large numbers of classes which can be grasped, such as bottle, banana, umbrella and teddy bear. Besides, the grasping dataset of self-supervised is used to train the pose estimation network and the dataset of Cornell University is used to validate the accuracy of pose estimation to ensure the novel of inference objects. Finally, the manipulator is used to grasp and place objects with the manipulator of Schunk.

A. RESULTS OF INSTANCE SEGMENTATION

The training of segmentation network has 180000 iterations and there are a lot of hyperparameters to tune during the training. The base learning rate is set to 0.02, which is relatively large and it decays with steps. Gamma of learning rate is 0.1 and decays at the 120000 and the 160000 iterations. Besides, weight decay is set to 0.0001.

We evaluate the instance segmentation network with the coco_val_2017 dataset, 5000 images included totally. By the reason that bounding boxes of detection and masks of segmentations are both used in the following experiment, we evaluate them separately. Our work is based on Faster R-CNN and we use the ResNet-101, a deeper network, with the feature pyramid network to predict the result, which seems a more accuracy result. We compare it with Faster R-CNN [27], YOLO v3 [35] and SSD513 [36]. The detection results are shown in Table 1, there is no doubt that our network is a good solution for detection because it improves more than 4.7% on mean average precision compared with the second-highest Faster R-CNN method. It shows that feature pyramid network helps an improvement to detect multi-scale objects.

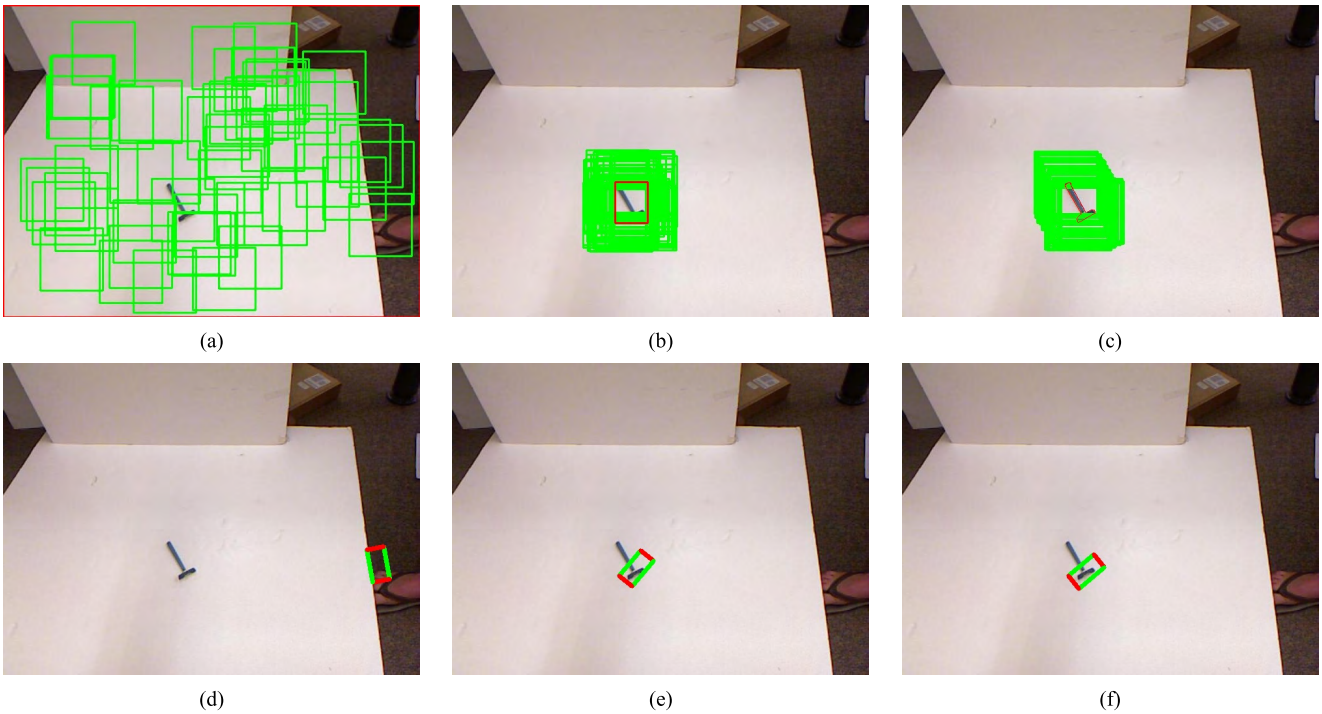


FIGURE 7. Patches and results in different sampling method. (a) shows the sampling range and patches of the method (1), (b) shows the sampling range and patches of the method (2), and (c) shows the sampling range and patches of the method (3). (d) shows the estimation result of method (1), (e) shows the result of method (2), and (f) shows the result of method (3).

TABLE 2. Segmentation results on COCO.

METHOD	DATA	AP, IOU:			AP, AREA:		
		0.5:0.95	0.5	0.75	S	M	L
MNC	VAL_2017	24.6	44.3	-	4.7	25.9	43.6
FCIS	VAL_2017	28.8	48.7	-	6.8	30.8	49.5
FCIS+ OHEM	VAL_2017	29.2	49.5	-	7.1	31.3	50.0
OURS	VAL_2017	36.4	58.5	38.7	16.6	39.2	54.0

Besides, segmentation has also been evaluated on the same dataset, as shown in Table 2. MNC [15] (Multi-task Network Cascades) and FCIS [18] (Fully Convolutional Instance-Aware Semantic Segmentation) are art-of-state methods in instance segmentation which detects and segments separately and they can be optimized with OHEM [37] (Online Hard Example Mining). From this table, we know that our segmentation network also answers well on segmentation with an improvement about 7.2% on mean average precision compared with the optimized FCIS thanks to the independence of masks between different classes.

Compared to former instance segmentation approach, our network shows great results on occlusion, which is common in object grasping environment. From FIGURE 6, it can be concluded that no more competition among classes is existed and spurious edges are disappeared. It costs 1.21 seconds for $1920 * 1080$ input image, which is the size of Kinect color frame. But if we resize the image into $640 * 480$, its inference time decreases to 0.12 seconds. Most of inference time are cost on up-sampling and the whole inference speed can fit in the requirement of real-time detection.

B. RESULTS OF POSE ESTIMATION

In these experiments, we set the step of grasping attitude θ to 10° as a tradeoff. If the step is too large, accuracy is not high whereas inference will cost too much time if the step is too small. We use the bounding box and the mask from instance segmentation as the heuristic input of the pose estimation network.

There are three different kinds of sampling methods mentioned as follows: (1) we sample patches from the whole image as inferred in [26], which means the center of patches should lie in the range of the whole image (2) we sample patches from the bounding box of the detected object, (3) we sample patches inside of contour of the object. (1) is the traditional way of sampling of the network and has the biggest sampling range. Method (2) and method (3) have much smaller sampling range and are easier to approach the accurate grasping position. We compare these three methods for the object be segmented in the former network. We sample the fixed quantity of sampled patches and compare the pose estimation accuracy.

In FIGURE 7, we show the patches and results in three methods to predict on a shaver. Sampled patches are set to 50 and it costs 0.15 seconds on the TITAN XP to predict the pose estimation for each image. In FIGURE 7, (a) to (c) show the corresponding fifty patches in method of (1) to (3) and

TABLE 3. Pose errors between different methods.

Approaches	Position-success (%)	Position error avg.± std(mm)	Attitude-success (%)	Attitude error avg.± std(deg)	Pose-Success (%)
Method (1)	42.36	12.77±6.36	50.06	21.97±7.25	23.22
Method (2)	88.30	11.18±6.12	59.57	19.15±6.49	51.06
Method (3)	97.37	9.37±5.27	59.77	20.01±6.61	58.27

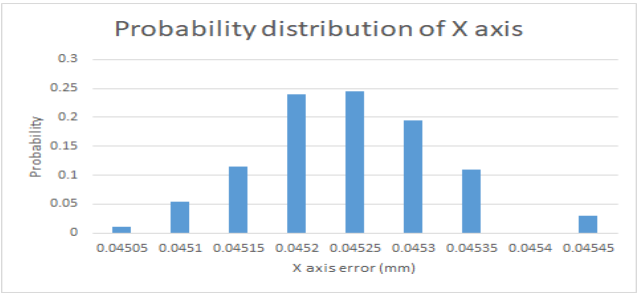
TABLE 4. Tradeoff between pose errors and inference time.

Sampled Patches	50	100	200	300	500	1000
Position error avg.± std(mm)	10.07±5.25	9.83±5.22	9.37±5.27	9.51±5.07	9.53±5.27	9.51±4.99
Attitude error avg.± std(deg)	20.14±6.92	20.61±6.85	20.01±6.61	20.05±6.69	19.98±6.52	19.82±6.68
Inference time(s)	0.15	0.26	0.75	1.01	1.44	2.83

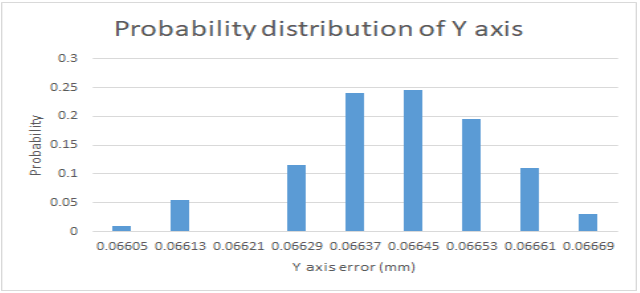
the red lines are the range of sampling positions. The method (3)'s sampling range is smallest and closest. It can be inferred that (2) and (3) is higher ranked than (1) in sampled patches. (2) and (3) are more likely to sample at the suitable position for grasping, which can be shown in (d) to (f). Figure (d) to (f) separately correspond to method (1) to (3). The red lines show the location of gripper. The gripper still has a distance when it is closed. As for shaver, its handle is too thin for manipulator to grasp. Network without self-supervised cannot deal with this problem. And (3) shows a higher success rate in grasping than (2) which is common when the object is thin, long and the object is placed with an orientation bias. In this situation, bounding box's area (area in red lines in (b)) is much bigger than the real area of the object (area in red lines in (c)). It shows a bigger difference in performance in Table 3 when encountering large validation dataset.

We validate our approaches using the grasping dataset of Cornell University. There are 833 images in total and it includes over 100 kinds of novel objects which have not disappeared in the training dataset. We set the sampling quantity to 200 for example and select the one with highest network output score as the predicting result. The difference between the predicting result and the ground truth is calculated as the pose error. We categorize the pose failure as exceeding errors of more than 30mms in position or 20 degrees in attitude compared with the ground truth.

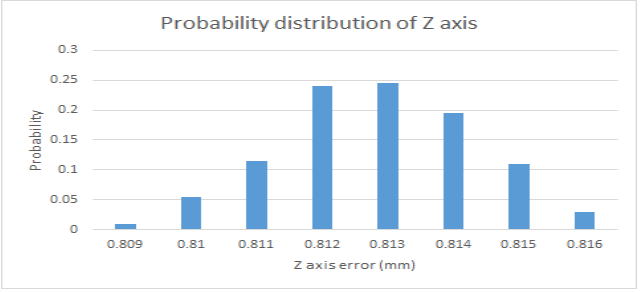
In Table 3, the method (2) and method (3) we proposed show an obvious improvement on position success. Thanks to instance segmentation, the method (3) based on it increases the position success to 97.37% and decreases the average position error to 9.37mm, which means it almost does not make mistakes when calculating the grasping position. As for attitude, the method (2) and (3) shows just a little better than the method (1) because they remove some incorrect attitude estimations of incorrect position calculations. The attitude success rate is still limited by the estimation network output size, so it cannot be as high as position success rate. Therefore, the instance segmentation has an indirect influence on attitude estimation. In general, the method (3) is 7.22% higher than the method (2) in the overall pose estimation success rate and it is 35.05% higher than the method (1). In a word, the heuristic knowledge based on instance segmentation inputted to estimation network helps a lot, which shows better performances than the original network and the network only based on detection. Therefore, we select the network with method (3) as the pose estimation part of our gasping approach.



(a)



(b)



(c)

FIGURE 8. Distribution of the pose transformation value.

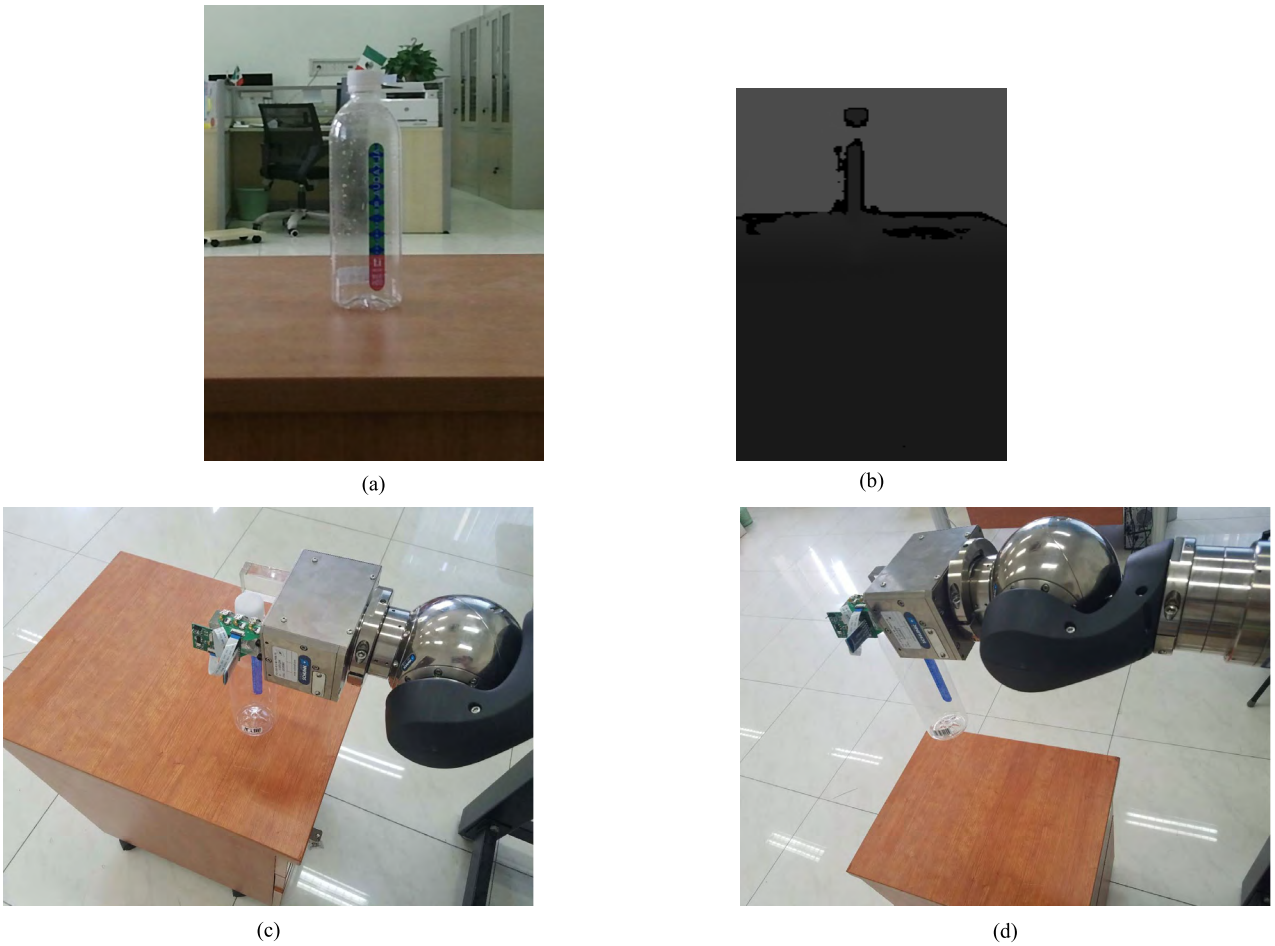


FIGURE 9. (a) is the color frame of Kinect after resizing and (b) is the corresponding depth frame. (c) shows that the manipulator has arrived the grasp pose and prepare to grasp. (d) shows the state that the object can be easily grasped.

Pose error includes position error and attitude error. Larger of sampled quantity leads to a lower pose error and a longer inference time. So, we want to select a best sampled patches quantity as the tradeoff between pose estimation error and inference time. We set the sampling quantity to 50, 100, 200, 300, 500 and 1000 separately and use the method (3) as the inference framework on the $640 * 480$ input image. Results are shown in Table 4. For 50 and 100 sampled patches, errors are relatively large and they can be decreased with the increasing of sampling quantity. As for 300, 500 and 1000, the pose inference time is more than 1 second, which seems to be unacceptable for real-time calculation. Besides, they also do not show obvious decrease in errors compared to the sampling quantity of 200. Thus, we choose the sampling quantity of 200 to estimate the grasping pose.

Besides, the whole segmentation and pose estimation process can be finished in 0.9 seconds for a resized $640 * 480$ input image and the sample quantity of 200 on TITAN XP, which is much quicker than other approach based on deep learning as 4 seconds in [25]. In another word, it can achieve the target of real-time calculation.

TABLE 5. Three-dimensional measurement and transformation value.

	Measurement Value(mm)	Transformation Value	
		Average(mm)	standard deviation(mm)
X axis	43.8	45.23	0.08
Y axis	64.9	66.36	0.12
Z axis (depth)	809.0	812.8	1.48

C. RESULTS OF POSE TRANSFORMATION

We transform to three-dimensional coordinates 200 times for a fixed pixel in the color frame and compare them with the measurement value. Results are shown in Table 5 and FIGURE 8, and these axes are illustrated under depth camera's space.

In Table 5, we can get the conclusion that the depth error is 3.8mm and errors in x and y axis is even smaller, which are in the range of 2mm. These errors can be acceptable for grasping. FIGURE 8 shows the probability distribution of the pose transformation value and we can see that they approximately conform to the normal distribution in three axes. Because the transformation value in X and Y axes is unchangeable when the value in Z axis is fixed, the distribution in three axes

presents a similar probability. Besides, transformation value in Z axis is not continuous with the step of 1mm.

D. RESULTS OF GRASPING

With the three-dimensional coordinates, we can grasp objects as we want. We take the transparent bottle as example for grasping. Tradition approach based on manual teaching cannot grasp an object in an arbitrary position and former network does not perform well on texture-less object. The manipulator's grasping procedure are shown in FIGURE 9. The upper figures in FIGURE 9 show the RGB frame and the depth frame of the grasping object in the view of Kinect. The RGB frame has a higher resolution than the depth frame and instance segmentation and pose estimation are based on the RGB frame. After the grasping pose is calculated, the three-dimensional coordinates can be transformed with the help of the depth frame. (c) in FIGURE 9 shows that the manipulator arrives the transformed pose and prepares to grasp. At this moment, the gripper of manipulator is at the maximum value of its move range. As shown in (d), the gripper closes and the manipulator rises after the pressure sensor on the gripper can feel the constant pressure for a few seconds. Thus, our whole approach also works well on real hardware grasping.

V. CONCLUSION

In this paper, we propose a new approach based on instance segmentation and self-supervised learning pose estimation network. These two networks are combined into a stable network. The output of instance of segmentation especially the contour of object can increase the pose estimation accuracy about 35% in the following network. Pose are transformed into three-dimensional coordinates under manipulator. Then the approach can grasp objects with the help of Schunk manipulator. During experiments of grasping evaluations on dataset, we can get the conclusion that our approach is a more accuracy approach for object grasping. And from experiments of grasping texture-less objects like transparent bottle on real hardware, we can find that our approach is a more robust for all kinds of objects.

ACKNOWLEDGMENT

Grateful acknowledgment is made to my supervisor Liu Chang who gave me a lot of helps.

REFERENCES

[1] P. R. Wurman and J. M. Romano, "The Amazon picking challenge 2015," *AI Mag.*, vol. 37, no. 2, pp. 97–98, 2016.

[2] Y. L. Cun *et al.*, "Handwritten digit recognition with a back-propagation network," in *Proc. Adv. Neural Inf. Process. Syst.*, Feb. 1990, vol. 2, no. 2, pp. 396–404.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, 2012, pp. 1097–1105.

[4] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vis. Res.*, vol. 37, no. 23, pp. 3311–3325, 1997.

[5] A. B. Rodney, "Planning collision- free motions for pick-and-place operations," *Int. J. Robot. Res.*, vol. 2, no. 4, pp. 19–44, Apr. 1983.

[6] K. B. Shimoga, "Robot grasp synthesis algorithms: A survey," *Int. J. Robot. Res.*, vol. 15, no. 3, pp. 230–266, Mar. 1996.

[7] T. Lozano-Perez, J. L. Jones, E. Mazer, and P. A. O'Donnell, "Task-level planning of pick-and-place robot motions," *Computer*, vol. 22, no. 3, pp. 21–29, Mar. 1989.

[8] V.-D. Nguyen, "Constructing force-closure grasps in 3D," in *Proc. IEEE. Int. Conf. Robot. Autom.*, Washington, DC, USA, Mar./Apr. 1988, pp. 3–16.

[9] X. Du, Y. Cai, T. Lu, S. Wang, and Z. Yan, "A robotic grasping method based on deep learning," *Proc. ROBOT*, vol. 39, no. 6, pp. 820–828, Jun. 2017.

[10] A. Zeng *et al.*, "Multi-view self-supervised deep learning for 6D pose estimation in the Amazon picking challenge," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May/Jun. 2017, vol. 39, no. 6, pp. 1383–1386.

[11] J. M. Wong *et al.*, "SegICP: Integrated deep semantic segmentation and pose estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vancouver, BC, Canada, Sep. 2017, pp. 5784–5789.

[12] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *Proc. Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science, Jun. 2014, vol. 8695, no. 6, pp. 297–312.

[13] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, Jun. 2015, pp. 447–456.

[14] J. Dai, K. He, and J. Sun, "Convolutional feature masking for joint object and stuff segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, Jun. 2015, pp. 3992–4000.

[15] J. Dai, K. He, and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, USA, Jun. 2016, pp. 3150–3158.

[16] S. Zagoruyko *et al.*, "A multi-path network for object detection," in *Proc. Brit. Mach. Vis. Conf.*, New York, NY, USA, 2016, pp. 1–12.

[17] Y. Li, H. Qi, J. Dai, X. Ji, Y. Wei, "Fully convolutional instance-aware semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, Jul. 2017, pp. 4438–4446.

[18] J. Dai, K. He, Y. Li, S. Ren, and J. Sun, "Instance-sensitive fully convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, 2016, pp. 534–549.

[19] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Barcelona, Spain, 2016, pp. 379–387.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Venice, Italy, Oct. 2017, pp. 2980–2988.

[21] R. Rios-Cabrera and T. Tuytelaars, "Discriminatively trained templates for 3D object detection: A real time scalable approach," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sydney, NSW, Australia, Dec. 2013, pp. 2048–2055.

[22] C. Yuan, X. Yu, and Z. Luo, "3D point cloud matching based on principal component analysis and iterative closest point algorithm," in *Proc. Int. Conf. Audio, Language Image Process.*, Shanghai, China, Jul. 2016, pp. 404–408.

[23] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[24] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE Int. Conf. Comput. Vis.*, Toronto, ON, Canada, Nov. 2012, pp. 2564–2571.

[25] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *Int. J. Robot. Res.*, vol. 34, nos. 4-5, pp. 705–724, 2015.

[26] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50 K tries and 700 robot hours," in *Proc. IEEE Int. Conf. Robot. Autom.*, Stockholm, Sweden, May 2016, pp. 3406–3413.

[27] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2015, pp. 91–99.

[28] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, Jun. 2017, pp. 936–944.

[29] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Santiago, Chile, 2015, pp. 1440–1448.

[30] D. Katz, A. Venkatraman, M. Kazemi, J. A. Bagnell, and A. Stentz, "Perceiving, learning, and exploiting object affordances for autonomous pile manipulation," *Auto. Robots*, vol. 37, no. 4, pp. 369–382, Dec. 2014.

[31] N. Passalis and A. Tefas, "Learning bag-of-features pooling for deep convolutional neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Venice, Italy, 2017, pp. 5766–5774.

[32] N. Passalis and A. Tefas, "Concept detection and face pose estimation using lightweight convolutional neural networks for steering drone video shooting," in *Proc. 25th Eur. Signal Process. Conf.*, Kos, Greece, Aug./Sep. 2017, pp. 71–75.
[33] Y. Guo, "Mean square exponential stability of stochastic delay cellular neural networks," *Electron. J. Qualitative Theory Differ. Equ.*, vol. 2013, no. 34, pp. 1–10, 2013.
[34] Y. Guo, "Global asymptotic stability analysis for integro-differential systems modeling neural networks with delays," *Zeitschrift angewandte Mathematik Physik*, vol. 61, no. 6, pp. 971–978, Dec. 2010.
[35] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," Tech. Rep., 2018.
[36] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, 2016, pp. 21–37.
[37] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, Jun. 2016, pp. 761–769.

XIN SHU received the B.S. degree in electronic and information engineering from Xi'an Jiao Tong University, Shaanxi, China, in 2016. He is currently pursuing the Ph.D. degree in microelectronics with the State Key Laboratory of Transducer Technology, Institute of Electronics, Chinese Academy of Sciences, Beijing, China.

His research interests include tactile sensors and the computer vision for Humanoid robot and manipulator.

CHANG LIU received the M.S. and Ph.D. degrees from the California Institute of Technology in 1991 and 1996, respectively.

In 1997, he became an Assistant Professor with a major appointment in the Electrical and Computer Engineering Department, and a minor appointment in the Mechanical and Industrial Engineering Department. In 2003, he was promoted to the position of Associate Professor with tenure. From 2007 to 2013, he was a Lifetime Professor with Northwestern University, Evanston, IL, USA. Since 2013, he has been the Director of the State Key Laboratory of Transducer Technology, Institute of Electronics, Chinese Academy of Sciences, Beijing, China. His research interests cover microsensors, microfluidic lab-on-a-chip systems, and applications of MEMS for nanotechnology. He has 13 years of research experience in the MEMS area, and has published 100 technical papers and three books.

Dr. Liu received the NSF CAREER Award in 1998. He was an Associate Editor of the IEEE Sensors Journal. He teaches undergraduate and graduate courses covering the areas of MEMS, solid state electronics, and heat transfer. In 2002, he was elected to the "Inventor Wall of Fame" by the Office of Technology Management, University of Illinois.

TONG LI received the B.S. degree in engineering mechanics from Beihang University, Beijing, China, in 2010, and the M.S. degree in mechanical and electronic engineering from the Beijing University of Posts and Telecommunications, Beijing, in 2016.

Since 2016, he has been a Research Associate with the Institute of Electronics, Chinese Academy of Sciences, Beijing. His research interests include advanced robot technology, tactile feedback control, integration of vision and tactile for the robot, and manipulator trajectory optimization.

Dr. Li's awards and honors include the Academic Fellowship (Beijing University of Posts and Telecommunications), the Best Graduate of Beihang University in 2010, and the best Ph.D. thesis honor of the Beijing University of Posts and Telecommunications in 2016.

CHUNKAI WANG received the B.S. degree in electronic science and technology from the Wuhan University of Technology, Wuhan, China, in 2016. He is currently pursuing the master's degree with the State Key Laboratory of Transducer Technology, Institute of Electronics Chinese Academy of Sciences, Beijing, China.

His research interest includes the fabrication and applications of tactile sensors in Humanoid robot nanocomposite materials.

CHENG CHI received the B.S. degree in electronic information engineering from Hunan University, Changsha, China. He is currently pursuing the Ph.D. degree with the University of Chinese Academy of Sciences, Beijing, China. His research is based within the Institute of Electronics, Chinese Academy of Sciences, under the supervision of Dr. C. Liu.

From 2017 to 2018, he was a Teaching Assistant at the MEMS Foundation, University of Chinese Academy of Sciences, Beijing. His research focus is information fusion of robotic vision and tactile sensation.