Active Learning-Based Grasp for Accurate Industrial Manipulation

Xiaokuan Fu, Yong Liu, *Member, IEEE*, and Zhilei Wang

***Abstract*— We propose an active learning-based grasp method for accurate industrial manipulation that combines the high accuracy of geometrically driven grasp methods and the generalization ability of data-driven grasp methods. Our grasp sequence consists of pregrasp stage and grasp stage which integrates the active perception and manipulation. In pregrasp stage, the manipulator actively moves and perceives the object. At each step, given the perception image, a motion is chosen so that the manipulator can adjust to a proper pose to grasp the object. We train a convolutional neural network to estimate the motion and combine the network with a closed-loop control so that the end effector can move to the pregrasp state. In grasp stage, the manipulator executes a fixed motion to complete the grasp task. The fixed motion can be acquired from the demonstration with nonexpert conveniently. Our proposed method does not require the prior knowledge of camera intrinsic parameters, hand-eye transformation, or manually designed feature of objects. Instead, the training data sets containing prior knowledge are collected through interactive perception. The method can be easily transferred to new tasks with a few human interventions and is able to complete high accuracy grasp task with a certain robustness to partial observation condition. In our circuit board grasping tests, we could achieve a grasp accuracy of 0.8 mm and 0.6°.**

***Note to Practitioners*—The research in this paper is motivated by the following practical problem. Manipulators on industrial lines can complete high accuracy tasks with hand-crafting features of objects. The perception is only used for object detection and localization. It is not flexible since the prior knowledge differs from tasks, which takes a long time to deploy in a new task. Besides, only well-trained experts are qualified to complete the deployment process. Our grasp method uses a convolutional network to estimate the motion for manipulator directly from images. The camera is mounted on the manipulator and can perceive the object actively. The training data set of the network is specific for different objects that can be automatically collected with a few human interventions. Our method simplifies the deployment process and can be applied in 3C industry (computers, communications, and consumer electronics) where the products upgrade frequently.**

***Index Terms*— Accurate grasp, active learning, manipulator.**

I. Introduction

MANIPULATORS are widely deployed on industrial lines with hand-crafting solutions. They are programmed to handle a lot of repetitive high precision tasks which greatly improve productivity. It is not flexible since sensor data are passively perceived. Manipulation combined with active object recognition is a more flexible and efficient approach that can be applied in unstructured environments [1]. With the development of 3C industry, traditional-automated manufacturing faces new challenges because of the variety of 3C products. In order to follow up the upgrade pace of products, it requires a more flexible grasp method with acitve perception that can be rapidly deployed.

Geometric methods combined with visual servo control utilize feedback information extracted from a vision sensor to control the motion of the manipulator. The image information includes geometric information such as points, lines, contours, and ROI [2], [3]. Although these methods can complete high precision grasp tasks, they are heavily relying on prior knowledge, such as camera intrinsic parameters, hand-eye transformation, manually defined features of objects, and grasp configuration. The knowledge is specific for each task and can hardly be transferred to the new tasks. When grasping a new object, it takes a long time to redefine features performed by robot experts.

Flexible reprogramming enables manipulators to make use of prior knowledge when deployed in new specific tasks. Task-level programming is a kind of flexible programming method which divides skills into three layers: the primitive layer, the skill layer, and the task layer [4]–[6]. The skills which contain offline specification and online teaching are parameterizable. When applying a known skill to handle a new object, it only needs to pilot manipulator to target locations, as well as some via-points with nonexperts. However, these methods need to acquire the prior knowledge of a set of objects in advance, and they can only simplify the deployment process in a limited range of tasks.

General learning-based grasp techniques are widely used in grasp synthesis in recent years, which have great generalization ability to deal with known, familiar, and even unknown objects. These methods concentrate on object representation and perceptual processing, such as object recognition or classification and pose estimation [7]. Fusion of multiple modal information for object recognition are studied to improve the robustness of manipulation [8]–[10]. Usually, a deep neural

Manuscript received October 15, 2018; revised December 14, 2018; accepted January 20, 2019. Date of publication February 25, 2019; date of current version October 4, 2019. This paper was recommended for publication by Associate Editor H. Liu and Editor H. Liu upon evaluation of the reviewers' comments. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1302003 and in part by the National Natural Science Foundation of China under Grant U1509210. *(Corresponding author: Yong Liu.)*

X. Fu and Z. Wang are with the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, China.

Y. Liu is with the State Key Laboratory of Industrial Control Technology, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, China (e-mail: yongliu@iipc.zju.edu.cn).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2019.2897791

network is used to learn grasp knowledge. Levine *et al.* [11] proposes a convolutional neural network (CNN)-based grasp predictor to choose an optimal grasp motion for end effector and can successfully grasp a variety of objects without any prior information of the camera, object features, or model. Although deep learning techniques can well handle the problem of robustly grasp objects under uncertainty, they focus more on the grasp success rate rather than the stability and accuracy of grasping, which are especially important in the industrial grasp tasks and cannot be ignored. Thus, a more feasible approach that can grasp the specific target object with high precision and has the ability to transfer into a novel object's grasp task with a few manual interventions is desired.

Addressing on reducing the deployment time and improving the grasp robustness in industrial grasp application, we propose a specific and active learning-based grasp technique that combines the accuracy of geometric feature-based methods and the generalization capability of learning-based techniques. Our method integrates the manipulation and perception. At each step, manipulator actively moves and perceives the environment. A deep convolutional neural network is used to regress the motion for the end effector. Similar to task-level programming, the grasp configuration such as target position is taught by nonexpert during the initial procedure. Besides, our method does not require the information of camera parameter or manually defined features. For each specific task, these information are learned by actively collecting its training data set. Compared to general learning-based grasp techniques, our method can complete the grasp tasks with a high precision of the industrial manipulation level. We evaluate the performance of our method in the circuit board grasp task. The translational and rotational errors of end effector can achieve 0.8 mm and 0.6°, respectively. This precision is qualified to general industrial grasp tasks.

The main contributions of this paper are as follows.

1) We propose an active learning-based grasp method, which can achieve high accuracy grasp only with few prior knowledges of the environment.
2) The data collection procedure of grasp training for varied targets is automatic with few manual interventions which can be implemented by nonexpert.
3) Our method is robust to the conditions that the initial states of grasp targets can only be partially observed and that the illumination of the environment is varied.

The rest of this paper is organized as follows. Section II reviews the related research works in grasp area. In Section III, we introduce the framework of our proposed grasp method and our implementation process in detail. Section IV compares our approach with traditional geometric feature-based grasp method under different conditions. Finally, we conclude in Section V.

II. Related Work

Manipulators have widely been applied in ground and in space [12], [13]. Grasping is an important area of all robot manipulation. The grasp methods can be categorized as geometrically driven and data driven [7].

A. Geometrically Driven Methods

Geometrically driven, also known as analytic methods, analyzes the shape of a target object and plans a suitable grasp pose, based on the four properties: dexterous, equilibrium, stability, and dynamic behavior [14]. Geometrical grasp methods register images of rigid objects to a known database of 3-D models, which typically involves segmentation, classification, and geometric pose estimation from 3-D pose cloud point as well as images and then index precomputed grasps. Typical visual servo control usually combines with geometric methods. Position-based visual servoing (PBVS) and image-based visual servoing (IBVS) are two archetypal visual servo control schemes [2]. The key task of PBVS is to estimate the pose of object based on camera intrinsic parameters and known geometric features such as point features [15], contours [16], ROI [17], and corner points. IBVS also requires to extract geometric features first such as point features [2], line features [18], and then establishes a dynamic control model between manipulator vision space and motion space. Others features such as the brightness of the whole image [19], gray histogram [20], and image Gaussian mixture model [21] are also used.

Although geometrically driven grasp methods can complete high precision grasp tasks and are widely applied in industry, the generalization ability is weak since these methods heavily rely on hand-crafting features or 3-D model of objects, as well as camera parameters. When applied in new tasks, it takes a long time to redeploy the manipulator.

B. Data-Driven Methods

Based on existing grasp experience, data-driven methods rely on sampling grasp candidates for an object and ranking them according to a specific metric. Recent advances in deep learning and reinforcement learning have performed well in image classification, object recognition, and robot motion planning. In grasping, deep learning techniques are used to segment the target object from clutter environment [22], [23], analyze the optimal grasp regions directly from images or point clouds [11], [24]–[27], learn grasp experience from simulation or physic trials [28]–[30].

Deep learning technique is used to estimate the pose of the target object and then rank the grasp candidates. A typical learning-based pose estimation grasp pipeline is first segmenting the image and point clouds with a CNN, then fitting the segmentation point clouds with a known 3-D model to estimate the pose of object using ICP algorithm, and finally, completing grasp task with an IK solver [22], [23]. Although these methods have better generalization ability and can grasp objects in clutter environment with a high success rate, they require as much prior knowledge as geometric-based method, such as camera parameters and 3-D model of objects.

There are also learning-based methods predicting grasp configurations directly from sensor input. These methods generate grasp candidates sampled from an image or point clouds first, and then rank grasp candidates with a specific metric, usually a CNN. There are several forms to represent grasp candidates. Reference rectangle [24]–[26] indicates potential grasp

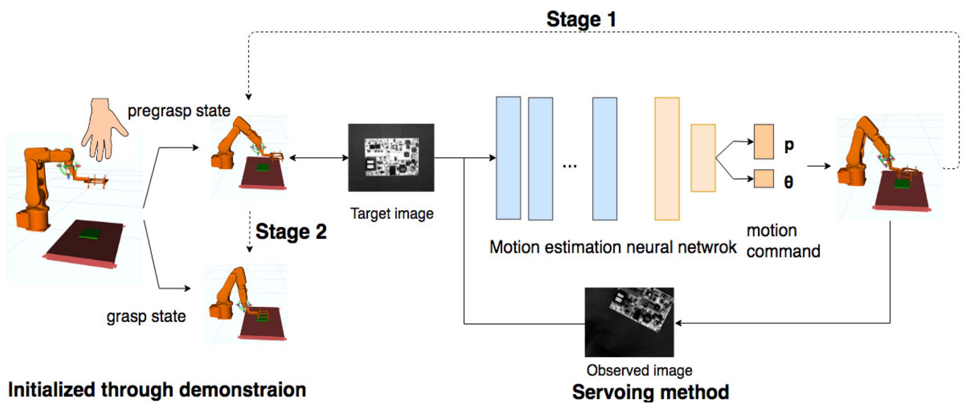


Fig. 1. Architecture: we divide the grasp task into two cascade grasp stages. The first stage is a deep learning-based servoing method where manipulator executes a motion command estimated by neural network and moves to the pregrasp state after several iterations. In the second stage, manipulator moves to the grasp state by executing a fixed command which is acquired through demonstration.

locations in the image. Motor commands are estimated with success probability directly from images [11], [31]. Grasp robustness of parallel-jaw grasp candidates from depth images are analyzed in [27]. Based on local features of objects, novel objects can be successfully grasped with these methods. However, network training requires large-scale data sets. Pinto and Gupta [24] spend 700 robot hours collecting a data set with a size of 50 k. Levine *et al.* [11] use 6–14 robots to collect a grasp data set with a size of 800 k over two months. It is expensive to acquire such a large grasp data set of the real manipulator. An alternative approach is generating synthetic data sets. Mahler *et al.* [27] build a synthetic data set of 6.7 million point clouds and trains GQ-CNN to analyze grasp robustness of candidates.

Although these methods can successfully grasp novel objects, they focus on the success rate of grasp rather than the stability or accuracy of grasp. For a novel object, it is difficult to estimate information on scales when using a monocular camera, which may cause some inaccurate grasps.

Another research trending on data-driven grasp methods is using deep reinforcement learning and imitation learning. These methods propose an end-to-end control solution in manipulation which learns to grasp objects by trial and error from large amounts of interaction data without any prior knowledge. The purpose is to train end-to-end visuomotor policies which could output the motor command directly from image input. These methods can learn a policy for solving a specific task, such as grasping a block [28]. However, it requires a large amount of interactions and sample data. Levine and Koltun [32], Levine and Abbeel [33], and Levine *et al.* [34] propose a method of guided policy search which greatly reduces the amount of sample data and improves the validity of the data. Imitation learning is a technique that learns control policies by observing demonstrations, which could teach robot to perform some complex tasks [29]. Though these methods have potential in generalization ability, they are only able to complete some simple tasks with a low precision. They cannot be applied in industrial grasp directly.

III. Our Two-Stage Grasp Method

Deep learning-based grasp techniques have achieved impressive success in recent years. However, these methods need to collect a large amount of data set and the precision of the grasp cannot qualify in industrial applications. In this section, we propose a two-stage grasp method based on active learning for accurate and stable industrial grasping tasks.

A. Overview of Our Two-Stage Method

In industrial robotic grasping, the target object may appear beyond the camera field of view when the tool or gripper approaches object. To resolve this problem, we propose an active learning-based two-stage grasp method which divides the grasp task into two grasp stages: pregrasp stage and grasp stage. The pregrasp state and grasp state are objective states corresponding to pregrasp stage and grasp stage. In pregrasp stage, manipulator moves to a proper position near to the target object. In grasp stage, manipulator executes a designed approaching motion command and completes the grasp task. Thus, our two-stage method can achieve a stable and high precision grasp and avoid collision with workbench. In our approach, we propose a deep learning-based servoing into our multistage grasp method. The architecture of our proposed method is shown in Fig. 1. The pregrasp state and grasp state are designed manually through demonstration, which determines a fixed approaching motion vector of the manipulator. Here, the motion vector represents the motion command for end effector, which contains a translational component and a rotational component. Since the implementation of the second stage is relatively simple, the primary aim of our method is to develop a servoing method that controls manipulator to move to the pregrasp state within limited iterations. Here, we use a convolutional neural network to estimate the motion command of end effector which takes raw images as input. The details of the first stage will be described in Section III-C.

B. Formal Definition for the Grasp Problem

In this section, we give formal definitions for the grasp task.

1) States: Let $x_i = (T_{\mathrm{ei}}, T_{\mathrm{ci}}, T_{\mathrm{oi}})$ denote the ith state of manipulator, camera, and object during a grasp process where T_{ei} specifies the frame of end effector, $T_{\mathrm{ci}} and T_{\mathrm{oi}}$ are the frames of camera and object. Let T_{ci}^{e} denote the transformation matrix from camera frame to end effector frame at state i. Our camera

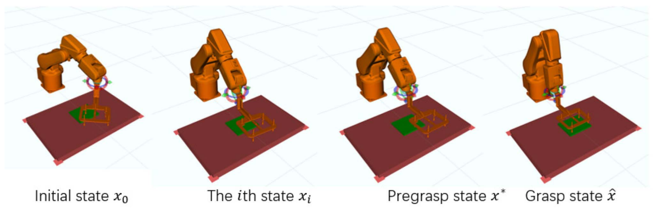


Fig. 2. Grasp sequence: x_0, x^*, and $\hat{x}$ are the three key states in a grasp sequence, x_i is an ordinary intermediate state. The green block represents the target object.

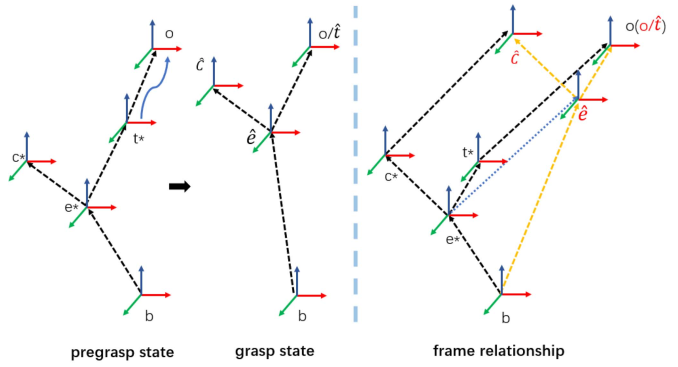


Fig. 3. Frame relationship between pregrasp state and grasp state: b, e, c, and t represent the base, end effector, camera, and tool of the manipulator. $x^* = (T_{c^*}, T_{e^*}, T_o)$ represents pregrasp state, while $\hat{x} = (T_{\hat{c}}, T_{\hat{e}}, T_o)$ represents grasp state. T_t is a tool frame. $T_{\hat{e}}^{e^*}$ represents the motion of end effector to transform from pregrasp state to grasp state.

is mounted on the end effector, thus, T_{ci}^{e} is a fixed matrix during a grasp process.

2) Image: Let I_i denote the observation by an RGB monocular camera corresponding to x_i. Let T_{oi}^{c} denote the transformation matrix from object frame to camera frame. I_i is determined by T_{oi}^{c} when camera intrinsic parameters are known.

3) Motion Command: We adopt a vertical grasp method. During grasping, the end effector is always perpendicular to the workbench, which can be applied in most industrial grasp applications. Here, we denote $v_t = (\mathbf{p}, \theta)$ as the motion command for the end effector, where $\mathbf{p}$ is the translational component specified in the frame of the manipulator's base and θ is the rotational component representing the relative change in rotation around the axis vertical to the target object.

4) Sequence: Let sequence $S = \{x_0, v_0, x_1, v_1, \ldots, x_i, v_i, x^*, v^*, \hat{x}\}$ denotes a grasp process, where x_i represents the state of manipulator, camera, and object after ith iterations. v_i is the motion command of end effector with respect to x_i, and manipulator transits to x_{i+1} after executing v_i. x_0 is the random initial state of manipulator system. x^* and $\hat{x}$ represent the pregrasp state and the grasp state. We can divide the sequence into S_1 and S_2 to describe the two stages, respectively. The whole grasp process is shown in Fig. 2.

In sequence S, $(x_0, x^*, \hat{x})$ are the three key states during grasping. In pregrasp state, the tool of the manipulator is close to the target object and the manipulator is ready to grasp the object. In the grasp state, the robot can complete the grasp task. Here, we use pregrasp pose and grasp pose to denote the coordinate of the end effector in pregrasp state and grasp state, respectively. For a specific object, we can always choose an optimal grasp configuration as the grasp pose utilizing our experience. In contrast, pregrasp pose is not uniquely determined. Let G represent the set of available pregrasp poses. We need to choose a proper $x^* \in G$, so that the manipulator system could transit to its related x^* from x_0 with minimal error. An empirical approach is choosing a pose just above the object with an appropriate height in z dimension so that the target appears in the center of the image and the end effector will move along a straight line without any rotation. This process is shown in Fig. 3. We have the relationship as

$$T_{c^*}^{e^*} * T_o^{c^*} = T_{\hat{e}}^{e^*} * T_{\hat{c}}^{\hat{e}} * T_o^{\hat{c}}. \tag{1}$$

Let T_y^x denote the coordinate transformation of frame y with respect to frame x, thus, $T_{\hat{e}}^{e^*}$ denotes the pose transform of end effector from pregrasp state to grasp state, which can be determined through demonstration.

5) Optimization Objective: We focus on optimizing the first stage process S_1 with a constraint criterion: the sequence S_1 should converge to x^* with a limited states. For the ith state, we have the relationship as

$$T_{c_i}^{e_i} * T_o^{c_i} = T_{e^*}^{e_i} * T_{c^*}^{e^*} * T_o^{c^*}. \tag{2}$$

We can derive the optimal motion vector for end effector is

$$T_{e_i}^{e^*} = T_{c_i}^{e_i} * T_o^{c_i} * T_{c^*}^{o} * T_{e^*}^{c^*} \tag{3}$$

where I_i is the observed image at state x_i, which contains information of $T_{\text{oi}}^{c_i}$. Since $T_{c_i}^{e_i}$, $T_{c^*}^{o}$, and $T_{e^*}^{c^*}$ are constant parameters after specifying a proper grasp state through demonstration, we can develop a function $f(I_i)$ which estimates the motion command v_t of end effector. We can optimize function f by minimizing a cost function $c[T_{e_i}^{e^*}, f(I_i)]$.

C. Implementation of the Pregrasp Stage

The objective of the first stage is to control manipulator to move to the pregrasp state within a limited time. The implement mainly consists of two components: the convolutional neural network to estimate motion command and the servoing method. Thus, we need to consider several issues.

1) The designed neural network should be qualified to regress relative pose with a high precision.
2) An efficient sample policy should be developed to reduce the number of physical samples during training as well as human interventions.
3) The network could be trained within an acceptable time interval.

We will then design the motion estimation network based on these above issues.

1) Network Architecture: Deep convolutional neural network performs well in the image process field. PoseNet [35] is a convolutional network regressing six-DOF camera pose directly from images which can obtain an accuracy approximate to 2 m and 6° in indoor scenes. Thus, CNN may be regarded as the form of nonlinear function to regress pose

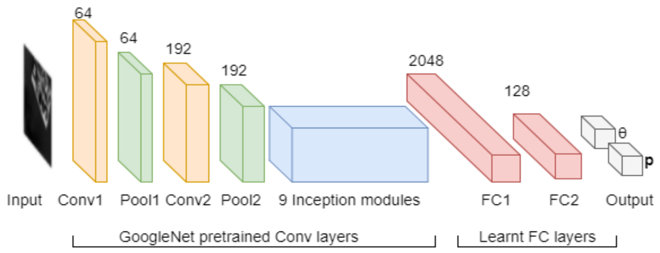


Fig. 4. Network architecture: we use pretrained convolutional layers of GoogleNet, followed with two fully connect layers to estimate the motion command.

TABLE I
COMPARISON OF AUTONOMOUS COLLECTING GRASP DATA SET

	Lerrel Pinto [24]	Sergey Levine [11]
Grasp attempts	50k	800k
Collecting time	700 robot hours	2 month/6∼14 robots

information. Similar to PoseNet, we propose our convolutional neural network on the basis of GoogleNet [36]. However, since industrial grasp task requires millimeter-level accuracy, we need to improve the pose estimation accuracy for the specific target object.

The input of our network is the current image I_t and the output is a command for end effector $\upsilon_t = (\mathbf{p}, \theta)$ which contains four dimensions. Considering the estimating error in z dimension is larger than x or y dimension for a monocular camera, we simply the output by cutting z dimension in order to improve the accuracy of our network. For each grasp task, the manipulator first moves to the same height as pregrasp pose, and then estimates command υ_t with the network where $\mathbf{p} = (x, y)$.

In addition, we modify the network structure by replacing softmax layers with regressors and insert two fully connected layers before regressor with the size of 2048 and 128, respectively. The regressor outputs a pose vector of three dimensions to represent the command of end effector. The architecture of our network is shown in Fig. 4. We first pretrain GoogleNet in image classification data set, and then transfer the convolutional layers of GoogleNet with trained weights to our motion estimation network directly.

2) Data Collection: The performance of the neural networks highly relies on the scale of the training data set. Although some state-of-the-art grasp networks are trained with autonomous collected data sets, shown in Table I, they still need too much time to deploy in industrial tasks. For the industrial grasp task, a more efficient and robust approach that can sample a relatively small data set and train the network within a few hours is especially important.

We propose a method that the manipulator can actively collect training data set with few manual interventions. For each new object, we only need to design a grasp state and a pregrasp state for the manipulator, then the manipulator will actively control the end effector and the camera will capture its perception as training data (the motion command and the corresponding image) for the training of the network.

Supposing image I_i is observed at the current state x_i, the output of network is $f(I_i)$, and the manipulator transfers to state x_{i+1} after executing a motion command $\upsilon_t = f(I_i)$. The optimal situation is that the sequence S_1 converges to x^* after only one iteration. Therefore, $T_{e^*}^{e_i}$ is an appropriate label for its corresponding image I_i.

When manipulator moves, the relative pose between object and camera changes, and the observed image also changes. We record the current pose of the end effector, and the label of each image is the relative translation vector to the pregrasp pose based on the current pose. In order to improve the efficiency of sampling, a proper sampling policy to cover the workspace of the end effector is necessary. In our approach, the Gaussian distribution- and the uniform distribution-based sampling policy are employed to efficiently cover the motion space. Our autonomous data collection process is shown in Algorithm 1. It is able to collect 10 k images in 6 h which can satisfy the precision training requirement in industrial grasp tasks.

Algorithm 1 Autonomous Data Collection Process
```
1: Determine pregrasp state $x^*$, where $T_{e^*}$ is the pregrasp pose
   of the end effector.
2: Training dataset $\mathscr{D} = \{\}$.
3: for $i < N$ do
4:     Choose a $T_e$ sampled from Gaussian distribution
       $\mathcal{N}(T_{e^*}, \Sigma)$ or Uniform distribution $\mathscr{U}(T_{e^*} - \sigma, T_{e^*} + \sigma)$.
5:     Control end effector to move to $T_e$, capture current image
       $I$, calculate its label $T_{e^*}^{e}$.
6:     Add $(I, T_{e^*}^{e})$ into $\mathscr{D}$.
7: end for
8: return $\mathscr{D}$.
```

3) Network Training: In industrial grasp task, it is critical to shorten the deployment time. Therefore, we pretrain the network in the image classification data set, and then fine tune with our collected data set, which could reduce the training time.

When designing the cost function, we need to balance the translation and the rotation errors. Here, we use Euclidean norm. Eqyations 4 and 5 present the loss function of the translational component and the rotational component, respectively, where $\mathbf{p}$ and θ are ground-truth values and $\hat{\mathbf{p}}$ and $\hat{\theta}$ are estimating values

$$\mathscr{C}_p(I_i) = \|\hat{\mathbf{p}} - \mathbf{p}\|_2 \tag{4}$$
$$\mathscr{C}_\theta(I_i) = \|\hat{\theta} - \theta\|_2. \tag{5}$$

We refer to [37] where learnable parameters $\hat{s}_p$ and $\hat{s}_\theta$ were introduced to balance the translational and the rotational errors. Then the final cost function is

$$\mathscr{C}(I_i) = \mathscr{C}_p(I_i)\exp(-\hat{s}_p) + \hat{s}_p + \mathscr{C}_\theta(I_i)\exp(-\hat{s}_\theta) + \hat{s}_\theta. \tag{6}$$

The size of the input image is 224×224, while the resolution of our camera is 1600×1200. We need to downsample the image to 400×300 first, and then crop it into 224×224.

The network is implemented on Tensorflow library. It is trained using stochastic gradient descent with the learning of 0.0001 and a batch of 75 on a single GPU(TITAN X Pascals). The whole training time is less than 1 h.

4) Servoing Method: Considering there will be errors on those regressed motion commands output by the trained network, it is difficult to control robot to move to the desired pose with just one iteration when using open-loop control process. We combine our network with a closed-loop servo control method. Our servoing method is presented in Algorithm 2. At each step, the network takes the inputs of currently perceived image I_t and estimates the motion command υ_t. We set a threshold for the estimated υ_t in order to reduce the oscillation of end effector. The manipulator will execute υ_t if υ_t is less than the threshold. In this method, the end effector could move to the desired pose after several iterations.

The threshold is set according to the precision of the motion estimation network. Here, we set the threshold to 0.8 mm in translational component and 0.6° in rotational component.

Algorithm 2 Servoing Method

```
1: Given current image $I_t$ and network $f$.
2: Compute the command $\upsilon_t$ using $\upsilon_t = f(I_t)$.
3: **while** $\upsilon_t > threshold$ **do**
4:     Control the end effector with command $\upsilon_t$, and capture
       the current image $I_{t+1}$.
5:     Compute the command $\upsilon_t$ using $\upsilon_t = f(I_{t+1})$.
6: **end while**
7: Control the end effector with $T_{\hat{e}}^{e^*}$.
```

D. Implementation of the Grasp Stage

In the second stage, manipulator executes an approaching translation vector to achieve the grasp task. It is a practical approach to avoid collision with the objects. The approaching translation vector is given through demonstration which is relatively simple and can be completed with nonexpert. The process is shown in Fig. 1.

IV. Experiments

A. Physical Evaluation Platform

Our experiments are performed on an ABB irb-120 manipulator, with a point grey FL3-GE camera mounted on its end effector. The grasp industrial object is a circuit board. Considering the goal of our deep learning-based grasp approach is to control the manipulator to move to the pregrasp pose from a random initial pose precisely, the precision of the second stage implement is determined by demonstration. Therefore, we do not evaluate the second grasp stage of our approach, which needs to equip a specific gripper. Our experimental hardware platform is shown in Fig. 5.

B. Data Collecting and Network Training

For network training, we hope to achieve the best training results with as few training samples as possible. Considering

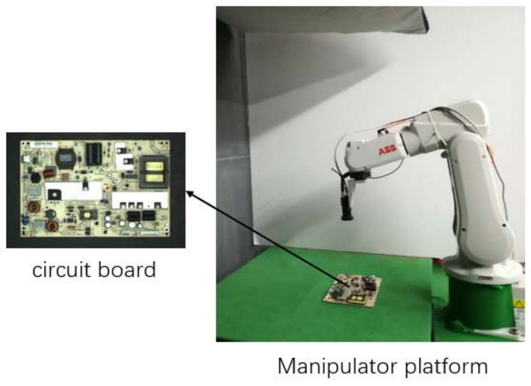


Fig. 5. Experiment hardware platform.

the amount of data samples, the optimal situation is to cover the sampling space with as few data as possible. We mainly consider the Gaussian distribution and the uniform distribution.

1) Uniform Distribution: All the samples in the sample space are equally important. It is a direct but effective sample method.

2) Guassian Distribution: The samples near to the pregrasp pose are more important than those far from pregrasp pose. This is mainly based on the situation that our servoing method needs several iterations to control manipulator to move to the target pose. When the pose of the end effector is far from target pose, it only needs to execute an approximate motion command so that it can move to a pose near to the pregrasp pose. On the contrary, the network needs to regress a precise motion command when the end effector is near to the target pose.

In order to compare the effects of different sample distributions on the experimental results, we collect a data set containing 10 044 samples with the Gaussian distribution and another data set containing 10 513 samples with the uniform distribution. For each data set, we select 9000 images as the training data set and the rest as the testing data set. Based on those two training data sets, we train two networks with the same architecture and compare their regression errors in their corresponding testing data sets.

The experimental results in Fig. 6 compare the translational error of the estimated motion command on different sample distribution data sets. Both sampled data sets are able to train a high precision motion estimation network where more than 95% samples in the testing data set achieve 5 mm precision in both x and y axes. When using the uniform distribution-based training data set, 48.3%(557/1153) of the testing data set can achieve less than 0.5 mm precision and 71.6%(825/1153) can achieve 1 mm precision, while only 9.1%(95/1044) of the testing data set can achieve 1 mm precision when using the Gaussian distribution-based training data set. Thus, the motion estimation network in subsequent experiments is trained with the uniform distribution data sets.

C. Grasping Results Analysis

Our servoing method is based on the motion command estimated with neural network. It is difficult to analyze the

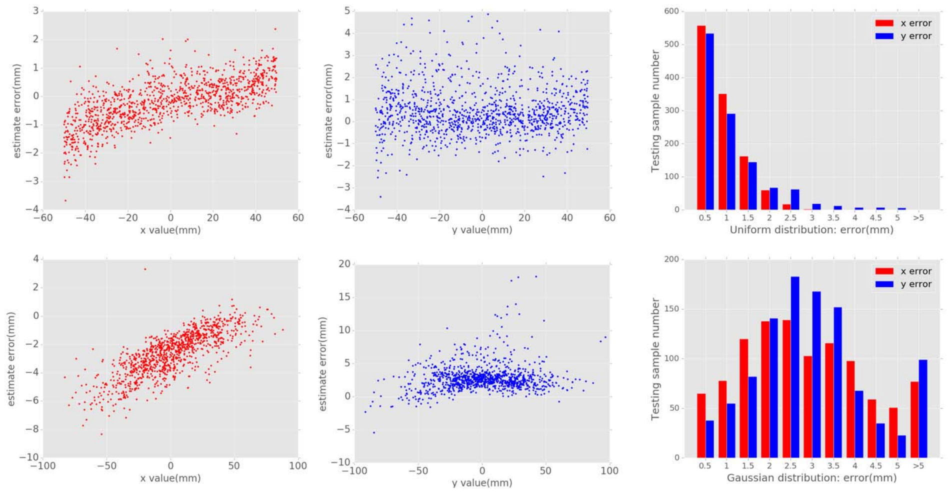


Fig. 6. Network regression error comparison between uniform distribution and Gaussian distribution data set: images in the first row are training results using Gaussian distribution data set, the second row are training results using uniform distribution data set.

convergence of the network. Here, we analyze the position error of our servoing method through several sets of experiments.

We take a known pregrasp pose as the ground truth so that we can compute the error of the network's output. In our experiment, we keep the same relative pose between the circuit board and the base of the manipulator. Under these circumstances, the pregrasp pose of end effector is uniquely determined. Then we randomly initialize the pose of end effector where the camera can observe at least a small part of the target object. Utilizing the proposed servoing method to control manipulator to move to the estimated pregrasp pose, we compute the error compared with the ground truth and evaluate our approach compared with the geometric feature-based methods. We present the baseline method as follows: point features are extracted to estimate the pose of object. Similar to our deep learning-based grasp technique, the manipulator will estimate its pregrasp pose according to those extracted features from observed images and move to the desired pose at each step. After several iterations, the manipulator can reach its pregrasping pose with small error.

To evaluate our proposed learning-based grasp method, we conduct three experiments. In the first experiment, we compare the performance of our method with the baseline method under the same initial conditions. In the second experiment, we adjust the intensity of illumination to test the robustness of our servoing method. In the last experiment, we change the initial pose of end effector where the circuit board can only be partially observed by the camera.

1) Comparison With Baseline Method: We conduct the first experiment to compare the performance of our deep learning-based grasp method with the geometric feature-based baseline method. This experiment includes ten trials where the pose of the target object is unchanged. These ten trials share the same pregrasp pose. In each trial, both methods start from the same random initial pose of the end effector, and we then compute the errors with the same target pose at the end of the pregrasp sequence $S1$. In each trial, our proposed method

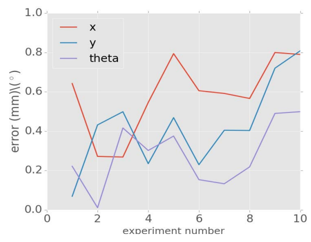
TABLE II
ERROR OF EACH STATE IN A GRASP SEQUENCE UNDER DIFFERENT ILLUMINATE CONDITIONS

	illuminate1			illuminate2		
iteration	$x(mm)$	$y(mm)$	$\theta(^\circ)$	$x(mm)$	$y(mm)$	$\theta(^\circ)$
0	-10.13	7.21	27.65	-10.13	7.21	27.65
1	-0.40	0.82	-0.09	-0.46	1.54	1.41
2	0.32	0.54	0.02	-0.12	0.47	0.73

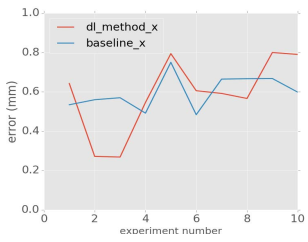
can complete $S1$ within two or three iterations. Fig. 7(a) shows the translational error and the rotational error of our proposed method are less than 0.9 mm and 0.6°, respectively. Fig. 7(b)–(d) compares the error of the end effector in each dimension when using our proposed method and the baseline method. In translational dimensions, our deep learning-based method performs as well as the baseline method. Although our deep learning-based method performs worse in the rotational dimension, its accuracy is still able to satisfy the requirements of industrial grasp.

2) Illumination: We test our method in two different illumination conditions with the same initial pose and target state. Fig. 8 shows the captured images of each state in a grasp sequence. The upper row is the grasp sequence under a weak intensity of illumination, while the bottom row is under a strong intensity of illumination. Table II shows the error of each state compare with the target state in the grasp sequence. The training data set of our motion estimation network is collected with a stable illumination condition, which means the changed illumination may not be included in our training data set. When the illumination intensity changes, our servoing method can still achieve an accuracy of 0.7 mm in translation and 0.8° in rotation.

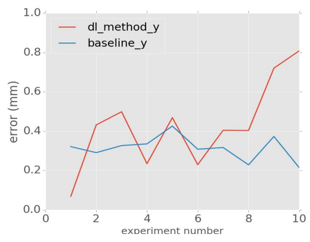
3) Partial Observation: Traditional geometric feature-based servoing method requires to extract features of the target object. It may fail to estimate the object's pose when part of the target object is beyond the camera field of view. We evaluate our method in such case. In our experiment, we initialize the



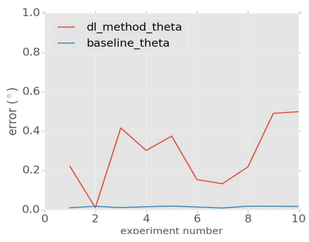
(a)



(b)



(c)



(d)

Fig. 7. Control errors comparision between deep learning-based method and baseline method. (a) Performance using deep learning method. (b)–(d) Comparisons between deep learning method and baseline method.

TABLE III
ERROR OF EACH STATE IN A GRASP SEQUENCE WHEN THE OBJECT OF INITIAL STATE IS PARTIAL OCCLUSION

	lower-right occlusion			lower-left occlusion			upper-left occlusion 1			upper-left occlusion 2		
iteration	$x(mm)$	$y(mm)$	$\theta(°)$	$x(mm)$	$y(mm)$	$\theta(°)$	$x(mm)$	$y(mm)$	$\theta(°)$	$x(mm)$	$y(mm)$	$\theta(°)$
0	-47.00	59.71	-39.21	-71.28	-94.72	-5.40	59.18	-123.13	-11.50	95.81	-0.64	-25.67
1	-1.92	2.48	-0.71	-5.48	-52.45	-4.36	-4.33	-72.97	-7.04	24.59	-16.68	2.46
2	0.26	0.20	0.21	-0.26	-1.72	1.56	-1.00	-26.26	-3.09	1.02	0.21	0.64
3	-	-	-	0.63	0.51	0.29	0.24	-0.24	-0.26	0.65	0.33	0.63

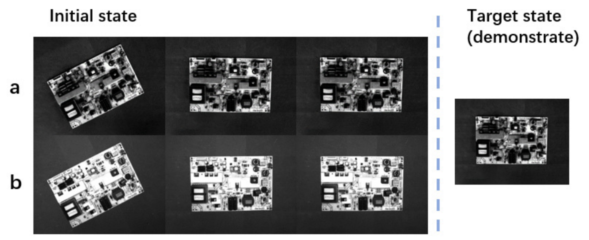


Fig. 8. Grasp sequence: observed images of each state under different illumination conditions.

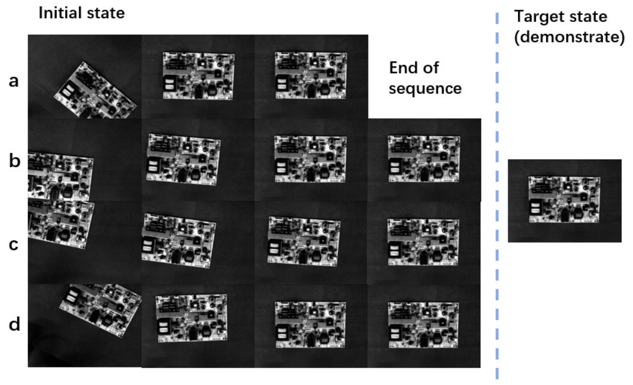


Fig. 9. Grasp sequence: observed images of each state when the object of initial state can only be partially observed.

end effector with a random pose where the target object is partial occlusion. Then, we control the manipulator to move to the pregrasp pose by our proposed method. As shown in Fig. 9, four grasp sequences with different partial occlusion initial states can converge to the target state within three adjustments. Table III shows the estimation error at each state during four grasp sequences. When the majority of the target object appears out of camera's view, there may still be a large error after executing the first estimated motion command. But the manipulator will move close to the pregrasp where the whole object can be observed. The overall error shows a decreasing trend.

V. CONCLUSION

This paper proposes a method integrating both deep learning technique and visual servoing for the purpose of solving industrial precise grasp tasks. Active perception is used in automatically and efficiently collecting the training data set for each specific object. Compared to traditional geometric feature-based visual servoing method, our method does not require to calibrate camera or extract features manually, so that the whole initialization process does not require too much manual interventions. In addition, our method is a precision stable and feasible grasp solution, which could be applied in industrial grasp compared with state-of-the-art deep learning-based grasp methods.

In the future, we will still further explore and improve our approach in the following aspects: such as introducing a more efficient intelligent active planning to further improve the sample efficiency, training the regressed motion of a full 6 degree of freedom, and adapting for the complex operating environment.

REFERENCES

[1] B. Calli, W. Caarls, M. Wisse, and P. P. Jonker, "Active vision via extremum seeking for robots in unstructured environments: Applications in object recognition and manipulation," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 4, pp. 1810–1822, Oct. 2018.

[2] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Trans. Robot. Autom.*, vol. 12, no. 5, pp. 651–670, Oct. 1996.

[3] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, Dec. 2006.

[4] R. S. Andersen, L. Nalpantidis, V. Krüger, O. Madsen, and T. B. Moeslund, "Using robot skills for flexible reprogramming of pick operations in industrial scenarios," in *Proc. IEEE Conf. Comput. Vis. Theory Appl. (VISAPP)*, vol. 3, Jan. 2014, pp. 678–685.

[5] M. R. Pedersen *et al.*, "Robot skills for manufacturing: From concept to industrial deployment," *Robot. Comput.-Integr. Manuf.*, vol. 37, pp. 282–291, Feb. 2016.

[6] C. Schou, R. S. Andersen, D. Chrysostomou, S. Bøgh, and O. Madsen, "Skill-based instruction of collaborative robots in industrial settings," *Robot. Comput.-Integr. Manuf.*, vol. 53, pp. 72–80, Oct. 2018.

[7] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—A survey," *IEEE Trans. Robot.*, vol. 30, no. 2, pp. 289–309, Apr. 2014.

[8] H. Liu, Y. Yu, F. Sun, and J. Gu, "Visual–tactile fusion for object recognition," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 996–1008, Apr. 2017.

[9] H. Liu, F. Sun, B. Fang, and F. Long, "Material identification using tactile perception: A semantics-regularized dictionary learning method," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 3, pp. 1050–1058, Jun. 2017.

[10] D. De Gregorio, R. Zanella, G. Palli, S. Pirozzi, and C. Melchiorri, "Integration of robotic vision and tactile sensing for wire-terminal insertion tasks," *IEEE Trans. Autom. Sci. Eng.*, to be published.

[11] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Robot. Res.*, vol. 37, nos. 4–5, pp. 421–436, 2018.

[12] Z. Meng, P. Huang, and J. Guo, "Approach modeling and control of an autonomous maneuverable space net," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 6, pp. 2651–2661, Dec. 2017.

[13] Y. Zhang, P. Huang, Z. Meng, and Z. Liu, "Precise angles-only navigation for noncooperative proximity operation with application to tethered space robot," *IEEE Trans. Control Syst. Technol.*, to be published.

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

[15] D. F. Dementhon and L. S. Davis, "Model-based object pose in 25 lines of code," *Int. J. Comput. Vis.*, vol. 15, no. 1, pp. 123–141, Jun. 1995.

[16] P. I. Corke and S. A. Hutchinson, "A new partitioned approach to image-based visual servo control," *IEEE Trans. Robot. Autom.*, vol. 17, no. 4, pp. 507–515, Aug. 2001.

[17] P. Huang, F. Zhang, J. Cai, D. Wang, Z. Meng, and J. Guo, "Dexterous tethered space robot: Design, measurement, control and experiment," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 3, pp. 1452–1468, Jun. 2017.

[18] N. Andreff, B. Espiau, and R. Horaud, "Visual servoing from lines," *Int. J. Robot. Res.*, vol. 21, no. 8, pp. 679–699, 2002.

[19] C. Collewet and E. Marchand, "Photometric visual servoing," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 828–834, Aug. 2011.

[20] Q. Bateux and E. Marchand, "Histograms-based visual servoing," *IEEE Robot. Autom. Lett.*, vol. 2, no. 1, pp. 80–87, Jan. 2017.

[21] N. Crombez, E. M. Mouaddib, G. Caron, and F. Chaumette, "Visual servoing with photometric Gaussian mixtures as dense features," *IEEE Trans. Robot.*, vol. 35, no. 1, pp. 49–63, Feb. 2019.

[22] 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. Autom. (ICRA)*, May/Jun. 2017, pp. 1383–1386.

[23] M. Schwarz, A. Milan, A. S. Periyasamy, and S. Behnke, "RGB-D object detection and semantic segmentation for autonomous manipulation in clutter," *Int. J. Robot. Res.*, vol. 37, nos. 4–5, pp. 437–451, 2018.

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

[25] D. Guo, F. Sun, T. Kong, and H. Liu, "Deep vision networks for real-time robotic grasp detection," *Int. J. Adv. Robot. Syst.*, vol. 14, no. 1, 2016, Art. no. 1729881416682706.

[26] D. Guo, F. Sun, H. Liu, T. Kong, B. Fang, and N. Xi, "A hybrid deep architecture for robotic grasp detection," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 1609–1614.

[27] J. Mahler *et al.*, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Proc. Robot. Sci. Syst.*, Cambridge, MA, USA, Jul. 2017, pp. 1–10.

[28] M. Andrychowicz *et al.*, "Hindsight experience replay," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5048–5058.

[29] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 5074–5082.

[30] A. A. Rusu, M. Večerík, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, "Sim-to-real robot learning from pixels with progressive nets," in *Proc. 1st Annu. Conf. Robot Learn.* ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78, Nov. 2017, pp. 262–270.

[31] R. Cheng, A. Agarwal, and K. Fragkiadaki, "Reinforcement learning of active vision for manipulating objects under occlusions," in *Proc. 2nd Conf. Robot Learn.*, vol. 87, Oct. 2018, pp. 422–431.

[32] S. Levine and V. Koltun, "Guided policy search," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2013, pp. 1–9.

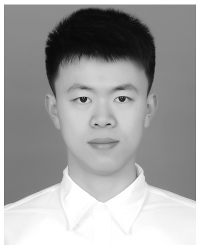
[33] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1071–1079.

[34] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 156–163.

[35] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2938–2946.

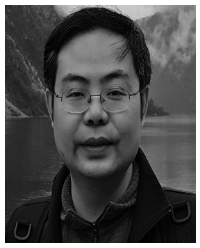
[36] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[37] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 3, Jul. 2017, pp. 5974–5983.



Xiaokuan Fu received the B.S. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2017, where he is currently pursuing the M.S. degree with the Department of Control Science and Engineering, Institute of Cyber-Systems and Control.

His current research interests include robotic manipulation, robotic vision, and reinforcement learning.



Yong Liu (M'11) received the B.S. degree in computer science and engineering and the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2001 and 2007, respectively.

He is currently a Professor with the Department of Control Science and Engineering, Institute of Cyber Systems and Control, Zhejiang University. He has published more than 30 research papers in machine learning, computer vision, information fusion, and robotics. His latest research interests include machine learning, robotics vision, information processing, and granular computing.

Zhilei Wang is currently pursuing the M.S. degree with the Department of Control Science and Engineering, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China.

His current research interests include robot grasping and machine learning.