applied sciences

MDPI

Article

Object Detection-Based One-Shot Imitation Learning with an RGB-D Camera

Quanquan Shao [1], Jin Qi [1], Jin Ma [2], Yi Fang [1], Weiming Wang [1] and Jie Hu [1,*]

[1] School of Mechanical Engineering, Shanghai Jiao Tong University (SJTU), Shanghai 200240, China
[2] School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), Shanghai 200240, China
* Correspondence: hujie@sjtu.edu.cn

Received: 27 November 2019; Accepted: 21 January 2020; Published: 23 January 2020

Abstract: End-to-end robot learning has achieved a great success for robots to obtain various manipulation skills. It learns a function which maps visual information to robotic action directly. Because of the diversity of target objects, most end-to-end robot learning approaches have focused on a single object-specific task with a limited capability of generalization. In this work, an object detection-based one-shot learning method is proposed, which separates the semantic understanding from robot control. It enables a robot to acquire similar manipulation skills efficiently and to have the ability to cope with new objects with a single demonstration. This approach mainly has two modules: the object detection network and the motion policy network. With RGB images, the object detection network tries to output the task-related semantic keypoint of the target object, which is the center of the container in this application, and the motion policy network generates the motion action based on the depth map and the detected keypoint. To evaluate this proposed pipeline, a series of experiments are conducted on typical placing tasks in different simulation scenarios and, additionally, the learned policy is transferred from simulation to the real world without any fine-tuning.

Keywords: object detection; one-shot imitation learning; sim-to-real; robotic manipulation

1. Introduction

Enabling robots to achieve the capability of performing all kinds of manipulation tasks is still a big challenge. In consideration of the diversity of manipulation tasks and surroundings, learning-based methods offer a promising generic paradigm to acquire these manipulation capabilities. With the development of deep learning methods, end-to-end robot learning approaches have been widely explored and achieved remarkable success for robots to solve a wide variety of robotic problems. These methods try to learn a function that maps the perception information to robot action directly, which can be regarded as an end-to-end visuomotor control manner. The perception information can be RGB images, depth maps, point clouds or other visual information. The robot action can be motion action and force/torque action. For both action types, they all could be relative changes or absolute values in a Cartesian coordinate space or joint coordinate space. It has been proved that end-to-end learning approaches have significant advantages than traditional methods in complicated manipulation tasks and in dynamic surroundings [1]. Nevertheless, these methods usually need lots of training data and their generalization capabilities are limited by the range of experience in the training phase [2]. Domain randomization is often used to diversify the training data to improve the performance for novel objects and new environments [3]. Leveraging previous learned skills to quickly learn new similar behaviors is also explored. Meta-imitation learning (MIL) learns a new policy via one or a few gradient steps with one or a few new demonstrations [4]. Task-embedded control networks (TecNets) use an embedding space to learn the similarities of different tasks, which could obtain a good performance in similar

new tasks [5]. Following these technical routes, an object detection-based framework is proposed for one-shot imitation learning, which is inspired by object-agnostic tracking method [6] and keypoint representation method [7].

The proposed framework mainly consists of two modules: the object detection module and the motion policy module. The object detection module first detects the target object with one shot demonstration. A cropped image of the target object and an image of the entire environment are inputted into a Siamese structure network to predict the semantic keypoint in image coordinates, which is the center of the target container. The motion policy module combines this semantic keypoint and the depth map of the environment to generate motion action. The position change of the end-effector and the state of the gripper are chosen as the motion action in this case. This framework separates the semantic understanding from robot control, which obtains a good generalization capability for novel objects and new environments. In addition, this framework not only could map the perception information to motion action, but also could generate different action in the same surroundings based on specific semantic keypoints and it is very important in robotic applications.

The main contributions of this proposed framework are twofold: (a) An object detection method is used for obtaining the image of the target object via a single demonstration, and further, the semantic keypoint is given with this template image and an image of the entire environment. (b) The semantic keypoint and feature blocks of depth information are integrated to generate motion actions, and this learned skill is easily transferred to novel objects and new environments.

The remainder of this paper is mainly organized as follows: In Section 2, some related studies about robot learning, imitation learning, object tracking and few-shot learning are discussed. Then the detailed structure of the proposed object detection-based one-shot imitation learning method is presented in Section 3. A series of experiments are conducted in simulation and the real world to validate the feasibility and performance of this approach in Section 4. Finally, conclusions and some future work suggestions are given.

2. Related Works

In this section, some related studies are reviewed, including robotic learning, visuomotor control, imitation learning, one-shot learning and object tracking. As a promising generic paradigm to acquire different kinds of manipulation capabilities, robot learning methods are widely explored [1,8,9]. Some studies apply learning strategies primarily in the perception or decision phase of robotic tasks and are integrated with traditional low-level control methods. Pinto et al. used a grasping detection network for object grasping and collected the training data in an online manner [9]. Zeng et al. proposed a self-learning method to grasp cluttered objects integrating the pushing and grasping action together [10]. Sui et al. applied a convolutional network to predict the position of the target in different environments [11]. As another framework, end-to-end learning-based approaches are also studied in different robotic manipulation tasks [12–15]. These methods try to learn a function that maps the perception information to robot action directly and these end-to-end learning approaches have significant advantages than traditional methods in complicated manipulation tasks and in dynamic surroundings [1]. These end-to-end visuomotor control methods are easily integrated with different learning framework, such as imitation learning (IL) and reinforcement learning (RL). Levine et al. combined end-to-end visuomotor control with a RL framework for a robot to learn contract-rich manipulation behaviors that is normally difficult to model using traditional methods [1,14]. Lee et al. applied visuomotor control method for peg-in-hole tasks by fusing perception information and proprioceptive information in a RL manner [16]. Kalashnikov et al. used a scalable RL framework for learning vision-based dynamic grasping skills [17]. However, it is required lots of exploration to obtain a skill via an RL manner integrated with visuomotor framework, which is regarded to be data-inefficient [18,19].

As a supervised learning framework, imitation learning method is also applied to learn the end-to-end visuomotor skills. There are two main types to exploit the labeled demonstrations in IL

framework. One is behavior cloning (BC), in which a robot learns a policy that maps the perception information to actions directly [19]. Another type is inverse reinforcement learning (IRL) [20], where a robot tries to learn a reward or value function with these given demonstrations. With this reward function, the robot could continue learning the skills in a RL manner. In this work, the author mainly discusses the first style. Zhang et al. applied end-to-end IL in different complex manipulation tasks and collected the training data with reality headsets, which validated the efficiency of IL [19]. Rahmatizadeh et al. used a visuomotor framework for multi-task manipulation with an inexpensive robot [21]. As mentioned above, the robustness and generalization of these end-to-end learning-based methods are limited by the range of experience in the training phase. Abolghasemi et al. applied a task-focused visual attention mechanism in the end-to-end IL framework to enhance the robustness of visuomotor manipulation skills [22]. James et al. trained a visuomotor skill for pick-and-place task in simulation with domain randomization and transferred the visuomotor skill to the real world without any fine-tuning [3]. Hämäläinen et al. applied an affordance detection method to compress task-related features and trained a visuomotor policy with these features to get a good generalization capability to new tasks and new objects [23]. Chen et al. used an adversarial feature to enhance the robustness of the end-to-end visuomotor skills, which is integrated with RL framework [24]. Nevertheless, these methods can only be applied in the situation with only one target object in the visual data. Few-shot imitation learning is studied to adapt the learned skill to novel objects and new environments [4,5]. Finn et al. proposed a meta-imitation learning (MIL) method to learn similar new behaviors via one or several gradient steps with new demonstrations [4]. Inspired by metric learning, few-shot learning methods in image process, James et al. applied the task-embedded control networks (TecNets) to learn the similarities of different tasks. It got a good performance in similar new tasks by combining the task-embedded vector with the visuomotor control framework [5]. Keypoint-based methods are also used for different robotic tasks to acquire more general manipulation skills [7,25].

In this paper, an object detection-based framework is proposed for one-shot imitation learning, and it can obtain a good performance for novel objects and new environments. This framework includes the object detection module and the motion policy module. Object detection module, which is inspired by visual object tracking methods, first detects the target object with a single demonstration. The cropped image of the target object and an image of the entire environment are inputted into a Siamese structure network to detect the semantic keypoint that is the center of the target container in image coordinates. Motion policy module combines this semantic keypoint and the depth map of the environment to generate motion action. Visual object tracking has been widely studied in the field of computer vision, which utilizes a class-agnostic object template to detect the location of this object in the query image. Bertinetto et al. applied a convolutional Siamese network for visual object tracking [26]. Li et al. improved this Siamese network by integrating the Region Proposal Network (RPN) [27] with Siamese networks [28]. In addition, Li et al. further extended this method with deeper networks and refined the cross-correlation layers for a better performance [6]. Shaban et al. predicted different weights for fusing two branches of Siamese network in one-shot semantic segmentation task [29]. A guided network is also used in few-shot semantic segmentation that is similar to visual object tracking [30]. In the proposed framework, an object detection framework is changed from [6] and applied to detect the semantic keypoint with template images. Although RGB images are mostly used as the perception information in visuomotor frameworks. The depth map is applied as the perception information and is combined with detected semantic keypoint to generate motion action in this framework. Compared to RGB images, depth maps have not color and texture information and have smaller gaps between simulation and the real world, which is beneficial for robots to acquire more general behaviors. Tai et al. used raw depth maps as inputs for visuomotor navigation [31]. Chen et al. trained the visuomotor policy with depth information and semantic information in simulation and transferred it to the real world for robotic navigation [32]. Morrison et al. learned grasping skills with synthetic depth maps and tested in the real world [33].

3. Methodology

The goal of the proposed method is enabling the robot to effectively interact with new, unknown objects in new environment from a single visual demonstration. In a BC framework, a policy $\pi(\boldsymbol{a}|\boldsymbol{o})$ that maps observations $\boldsymbol{o}$ to actions $\boldsymbol{a}$ is learned with demonstrations generated by expert policies π^*. A demonstration trajectory is composed of a series of observations and actions, which is formalized as $\tau = [(\boldsymbol{o_1}, \boldsymbol{a_1}), \ldots, (\boldsymbol{o_N}, \boldsymbol{a_N})]$. Trajectories of a task is formalized as $\Gamma^i = [\tau_1, \ldots, \tau_K]$ and trajectories of different tasks $[\Gamma^1, \ldots, \Gamma^L]$ are collected for training the policy. To cope with new, unknown objects, TecNets [5] applied a task-embedding processing to find the similarity with the first frame $\boldsymbol{o_1^d}$ and last frame $\boldsymbol{o_N^d}$ of the new demonstration. The embedded vector was called as a sentence $\boldsymbol{s}\left(\boldsymbol{o_1^d}, \boldsymbol{o_N^d}\right)$ and the policy generated different actions based on the task sentence, which is modulated as $\pi(\boldsymbol{a}|\boldsymbol{o}, \boldsymbol{s})$. Following this pipeline, the semantic keypoint $\boldsymbol{I}$ is applied to transfer specific intentions and the visuomotor policy is formalized as $\pi(\boldsymbol{a}|\boldsymbol{o}, \boldsymbol{I})$ where $\boldsymbol{I}\left(\boldsymbol{o}, \boldsymbol{o_1^d}, \boldsymbol{o_N^d}\right)$ is a function of observations $\boldsymbol{o}$, the first frame $\boldsymbol{o_1^d}$ and last frame $\boldsymbol{o_N^d}$ of the new demonstration. Because of the good generalization performance, depth map *s* $\boldsymbol{d}$ are used for the visuomotor policy and RGB images $\boldsymbol{o}$ are applied for keypoint detection. The visuomotor policy is formalized as $\pi(\boldsymbol{a}|\boldsymbol{d}, \boldsymbol{I})$ where $\boldsymbol{I}$ is an abbreviation of $\boldsymbol{I}\left(\boldsymbol{o}, \boldsymbol{o_1^d}, \boldsymbol{o_N^d}\right)$. Any other trajectory in the same task can be regarded as the target demonstration in the training phase. As a result, the robot could effectively interact with new, unknown objects in new environment from a single visual demonstration in the test phase.

Detailed structure of the proposed object detection-based one-shot imitation learning framework is shown in Figure 1. It mainly includes two parts: the object detection networks $\boldsymbol{I}\left(\boldsymbol{o}, \boldsymbol{o_1^d}, \boldsymbol{o_N^d}\right)$ and the motion policy networks $\pi(\boldsymbol{a}|\boldsymbol{d}, \boldsymbol{I})$. The perception data in robotic applications is normally various and high dimensional. Convolutional neural network (CNN) is often used to process this perception information. For both RGB images and depth maps, CNN-based encoder networks are used to extract image features. These encoder networks are often pretrained to improve of the learning speed of the robot learning system. In this framework, they are trained with an autoencoder (AE) structure firstly and then fixed in the robot learning phase. Object detection networks and motion policy networks are trained separately. The trained object detection networks output the center of the task-related container, which is regarded as the semantic keypoint. The policy networks integrate the compressed depth feature blocks and the semantic keypoint to generate motion action. The structure of the autoencoder networks and other two networks are introduced successively.

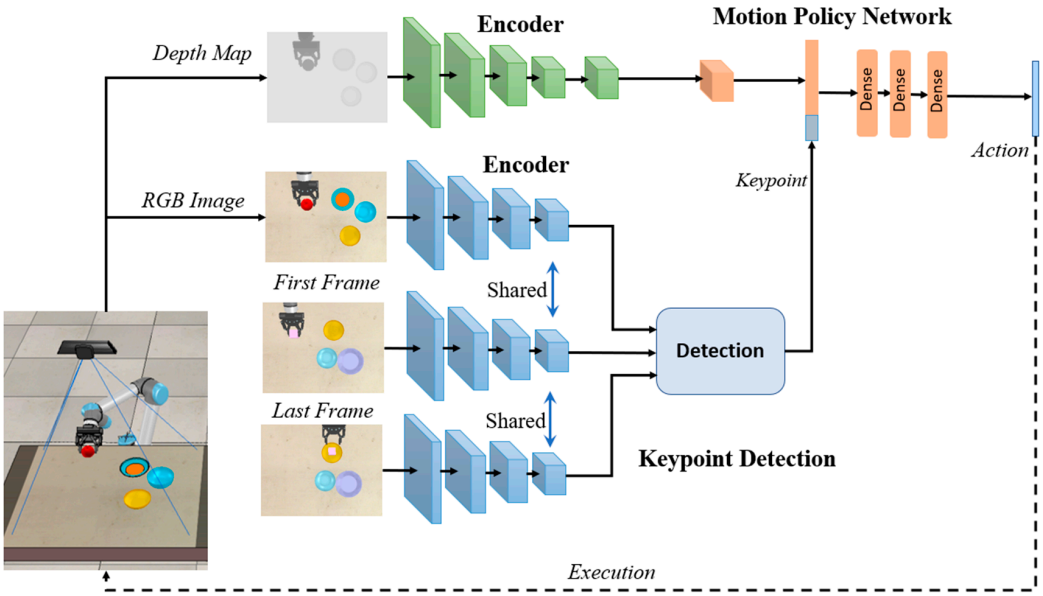

Figure 1. The main framework of the proposed method.

3.1. Autoencoder Network

AE is an unsupervised learning method and includes an encoder network and a decoder network. With the same inputs and outputs, AE can be trained in a self-supervised manner. The trained encoder network of AE can be used to process the perception information, which is RGB images and depth maps in this application. Two separated AEs are used to process different kinds of information. The detailed structures of these two AEs are similar and the last layers of these AEs have three channels and one channel respectively, as shown in Figure 2. The Rectified Linear Unit (ReLU) function is chosen as activation function and batch normalization is applied before the nonlinear process. Loss function of AE is normally the mean squared error of the inputs and outputs, which can be formalized as:

$$\mathcal{L}_{AE} = \|D(E(\boldsymbol{o})) - \boldsymbol{o}\|_2^2 \tag{1}$$

where $\boldsymbol{o}$ is the perception information that is an RGB image and a depth map in this situation. $D(*)$ and $E(*)$ are the decoder network and the encoder network respectively. L2 regularization is also used to avoid overfitting and the loss function is changed as:

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{AE} + \lambda_2 \mathcal{L}_{reg} \tag{2}$$

where $\mathcal{L}_{reg}$ is the loss of regularization and the hyper-parameters λ_1, λ_2 are the weights of different terms.

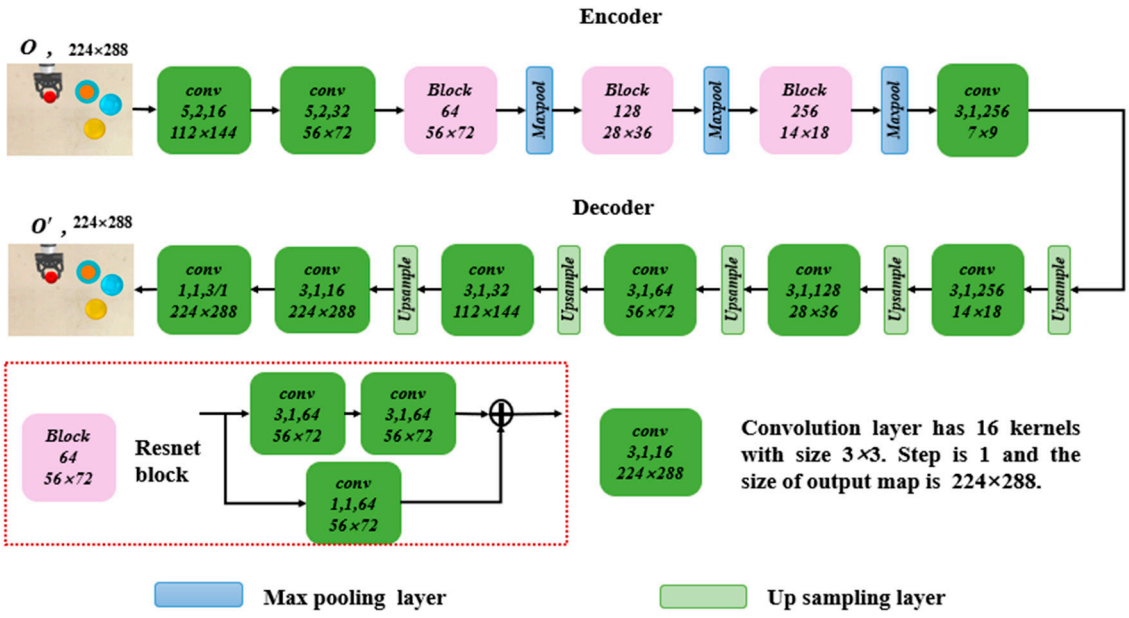

Figure 2. The detailed structure of autoencoder.

3.2. Object Detection Network

The output of object detection network is a probability map of the task-related semantic keypoint that is the center of the container in this case. The pixel with maximum value in this map is chosen as the semantic point and its pixel coordinates are inputted into the motion policy network. For one-shot learning framework, a demonstration trajectory is given. The first frame and last frame of the example trajectory are used to detection the task-related object, which is similar as [5]. Inputs of this object detection network are the RGB image of the whole environment, the first frame and last frame of the demonstration. Detailed structure of this network can be found in Figure 3. Trained encoder network is used to extract features of these images and these convolutional layers are fixed in the training phase. With the first frame and the last frame, the location of the task-related object is detected. Based on this location, two kernels are obtained by cropping the feature blocks of the start image with window

of size 3×3 and window of size 5×5. Two deep-wise cross correlation layers [6] are used for fusing these two kernels and the feature blocks of the environment respectively. After processes of some convolutional layers and deconvolutional layers, a probability map of the center of the container can be obtained. In other words, the target object is obtained firstly in the demonstration images and then exploit the feature of the target image to detection its location in the query image. This network is trained in two stages. The network is firstly trained to find the target object with the first and last frame. The prediction of this probability map is regarded as a classification problem and the cross-entropy loss is chosen as the loss function, which can be formulized as:

$$\mathcal{L}_1 = -\boldsymbol{M_L} log(\boldsymbol{M}) - (1 - \boldsymbol{M_L}) log(1 - \boldsymbol{M}) \tag{3}$$

where $\boldsymbol{M}$ is the predicted map and $\boldsymbol{M_L}$ is the labeled map. In the labeled mask, the value of the pixels less than three pixels from the center of the container is set to one and the others are all zeros. L2 regularization is used in this training phase and the loss function is changed as:

$$\mathcal{L}_{dt} = \lambda_3 \mathcal{L}_1 + \lambda_2 \mathcal{L}_{reg} \tag{4}$$

where the hyper-parameters λ_3, λ_2 are the weights of different terms. For the second training stage, the whole network is trained to predict the center of the container in the query image. Loss function is just same as Equations (3) and (4). After trained, the object detection network can output a probability map of the center of the container in the query image.

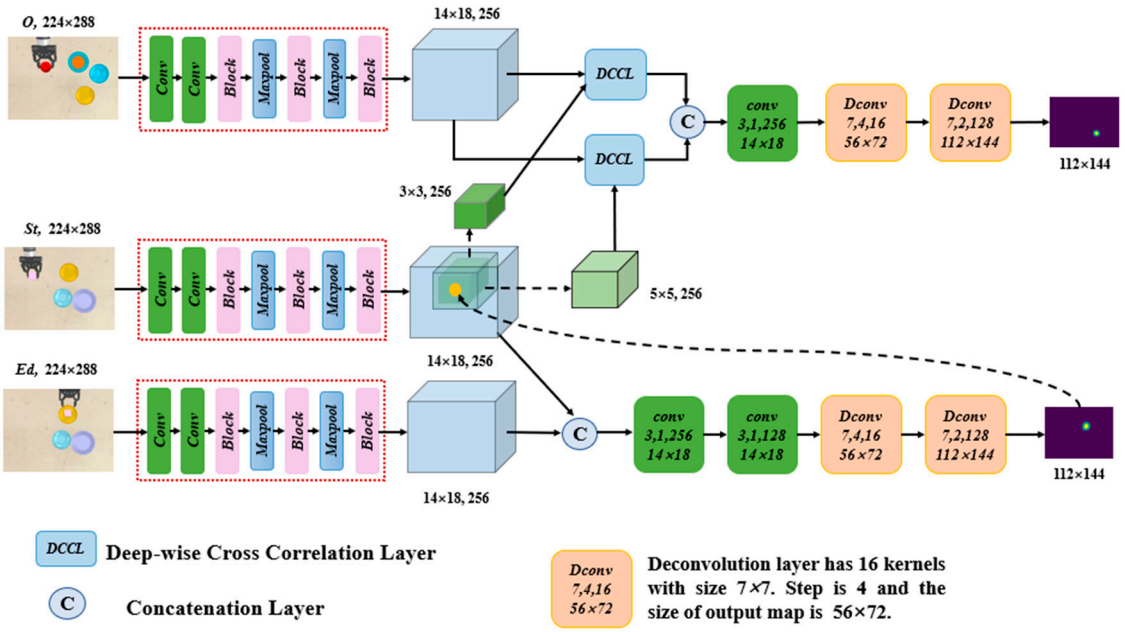

Figure 3. The structure of the object detection network.

3.3. Motion Policy Network

A depth map is chosen as the input of the motion policy network. The position change of the end-effector $\boldsymbol{P}$ and the state of the gripper $\boldsymbol{G}$ are made up of a motion action, which is the outputs of the policy network. The state of the gripper is a Boolean variable and the position change of the end-effector is a continuous variable. Therefore, the output layer of the position action is linear function and sigmoid function is used for the gripper state action. Detailed structure of the motion policy network is shown in Figure 4. The depth map of the environment is first processed by the trained encoder network and then these feature blocks are fed into a convolutional layer to compress their dimensions. Output of the convolutional layer is reshaped into a vector and concatenated with the

pixel coordinates of the keypoint that obtained by the object detection network. Two fully-connected layers are followed. With different output layers, the position action and the gripper action can be acquired. The activation function of these two dense layers is ReLU function and the dropout layer is also used between each dense layer to avoid overfitting. For continuous output, the mean square error is normally used as the loss function and cross-entropy function is the loss function of Boolean variable. Losses of the position change of the end-effector and the state of the gripper can be formulized as:

$$\mathcal{L}_p = \|\boldsymbol{P} - \boldsymbol{P'}\|_2^2 \tag{5}$$

$$\mathcal{L}_g = -\boldsymbol{G_L} log(\boldsymbol{G}) - (1 - \boldsymbol{G_L}) log(1 - \boldsymbol{G}) \tag{6}$$

$$\mathcal{L}_T = \lambda_4 \mathcal{L}_p + \mathcal{L}_g \tag{7}$$

where $\mathcal{L}_p$ and $\mathcal{L}_g$ are losses of position changes and the gripper states respectively. $\boldsymbol{P}$ is the labeled position change and $\boldsymbol{P'}$ is the predicted value. $\boldsymbol{G_L}$ is the labeled gripper state and $\boldsymbol{G}$ is the output of the network. Parameter λ_4 is the weight of loss of position change to increase the effect of this term.

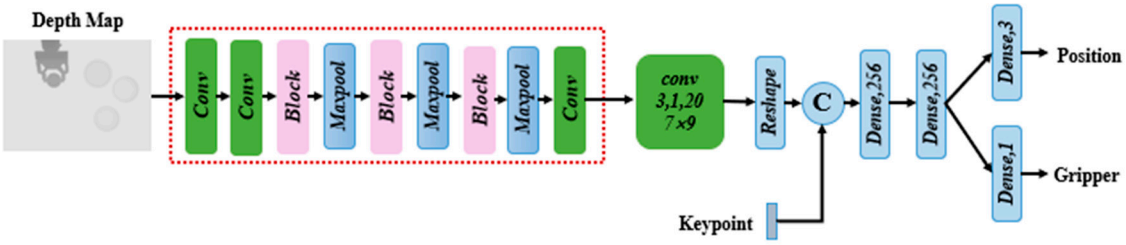

Figure 4. The structure of the motion policy network.

4. Experiments and Results

In order to validate this proposed object detection-based one-shot imitation learning framework, a series of simulated experiments are conducted in an object placing application scenario in the V-REP environment, shown in Figure 5. In the object placing task, the robot is guided by an RGB-D camera to place the object into the target container in the presence of two distractors on the table. The simulated RGB-D camera outputs an RGB image of size 240×320 and a depth map with the same size. The manipulator is a simulated UR5 robot equipped with a two-finger gripper and two grippers are used in the data collection phase to increase the diversity, which can be found in Figure 5. A workstation with a NVDIA GTX 1080Ti GPU and the machine learning platform of Tensorflow 1.70 are used for training. Twenty containers and twenty objects are used for collecting the training data and four cameras that are placed vertically above the table in different positions are applied for recording perception information at the same time. Containers and objects are shown in Figure 6. 1200 manipulation trajectories are collected and there are total 4800 demonstrations recorded by four cameras. 4000 samples are used as the training data and 800 trajectories are chosen for validating. A manipulation trajectory consists of a series of perception information $(\boldsymbol{o}, \boldsymbol{d})$ and motion actions $\boldsymbol{a} = (\boldsymbol{P}, \boldsymbol{G})$. As mentioned before, the position change of the end-effector $\boldsymbol{P}$ and the state of the gripper $\boldsymbol{G}$ are chosen as the action. With these demonstration trajectories, the visuomotor policy can be trained in a BC manner. Experiments are conducted to address three questions as followed: (1) Can the robot finish the object placing task efficiently for unknown containers in new environment with only one demonstration under this object detection-based framework? (2) Do the integrated kernels in the object detection module improve the performance of this approach and what is the effect of different kernel size? (3) Can this proposed framework trained in simulation be transferred to the real world directly without any fine-tuning?

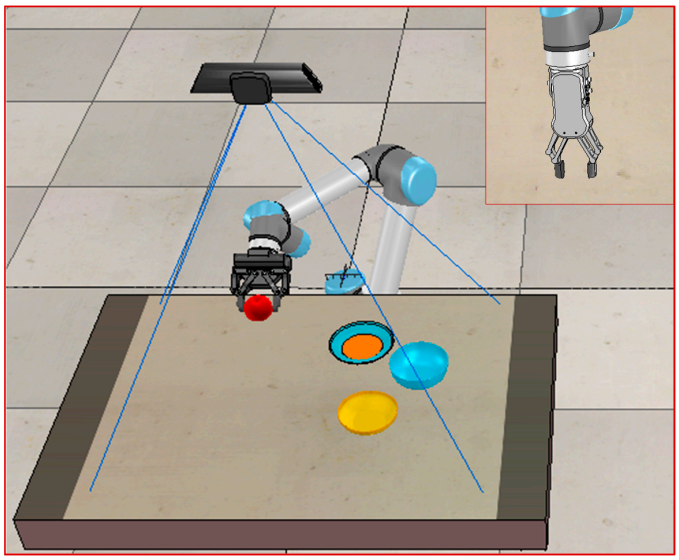

Figure 5. The object placing application scenarios in V-REP.

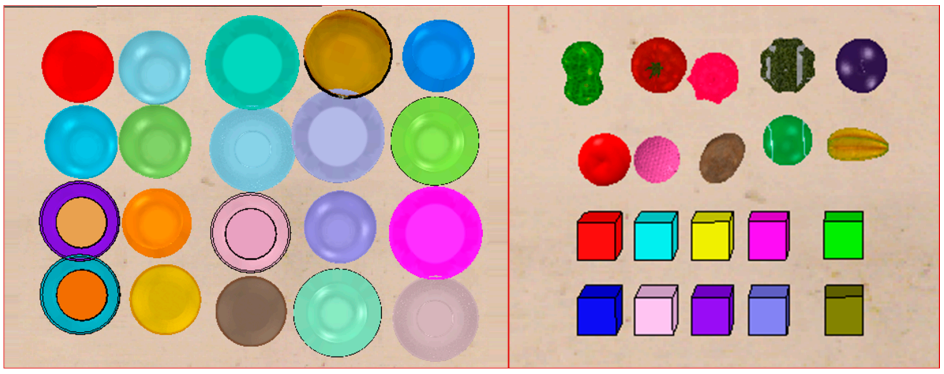

Figure 6. The objects and containers in V-REP.

4.1. Training the AE Networks

As mentioned above, two AEs are applied separately to process RGB images and depth maps. The size of the image got by the simulated camera is 240×320 while the input size of AE networks is 224×288. Initial images are randomly cropped to match the input size, which can also increase the diversity of training data.

Data augmentation methods including adding salt & pepper noise, flipping and rotating 180 degrees are also applied in the training phase. Adam optimizer is applied for training the network and related parameters are given in Table 1. The learning rate is 2×10^{-4} at beginning and is changed to 2×10^{-5} after 50 epochs.

Table 1. Parameters of different network module in the training phase.

	Networks	AE	Object Detection	Motion Policy
1	λ_1	1000	–	–
2	λ_2	10^{-5}	10^{-5}	–
3	λ_3	–	500	–
4	λ_4	–	–	1×10^4
5	(Adam) β_1	0.9	0.9	0.9
6	(Adam) β_2	0.999	0.999	0.999
7	Learning Rate	2×10^{-4}, 2×10^{-5},	2×10^{-4}	2×10^{-4}
8	Batch Size	64	16	64
9	Epochs	100	–	–
10	Iterations	–	20 K	30 K

4.2. Training the Object Detection Network

The object detection network is trained in two stages. The network is trained firstly to find the target object with the first and last frame in each demonstration trajectory. Then the whole network is trained to predict the center of the container in the query image. The trained encoder is used to process images into feature maps and they are fixed in this training phase. The aforementioned data augmentation tricks are also applied in this training phase of the object detection network. In addition, three channels are chosen randomly from the red channel (R), green channel (G), blue channel (B) and the gray channel (Gr) of RGB images, shown in Figure 7, which can dramatically increase the diversity of the perception information. To transfer the proposed framework from simulation to the real world, another 100 real images are also used in this training phase to reduce the sim-to-real gap of RGB images. Serval containers are placed on the table and an object is randomly place into one container. These real images are all labeled manually. Adam optimizer is also used for the training of object detection network. Hyper-parameters, learning rate and iterations are provided in Table 1. The lower branch in the object detection network is trained for 20 K iterations and then the up branch of this network is also trained for 20 K iterations. With the image of the environment and the first frame and the last frame of an example trajectory, the trained object detection network can output a probability map of the center of the target container. The pixel coordinates of the maximum value in this map is chosen to be inputted into the motion policy network.

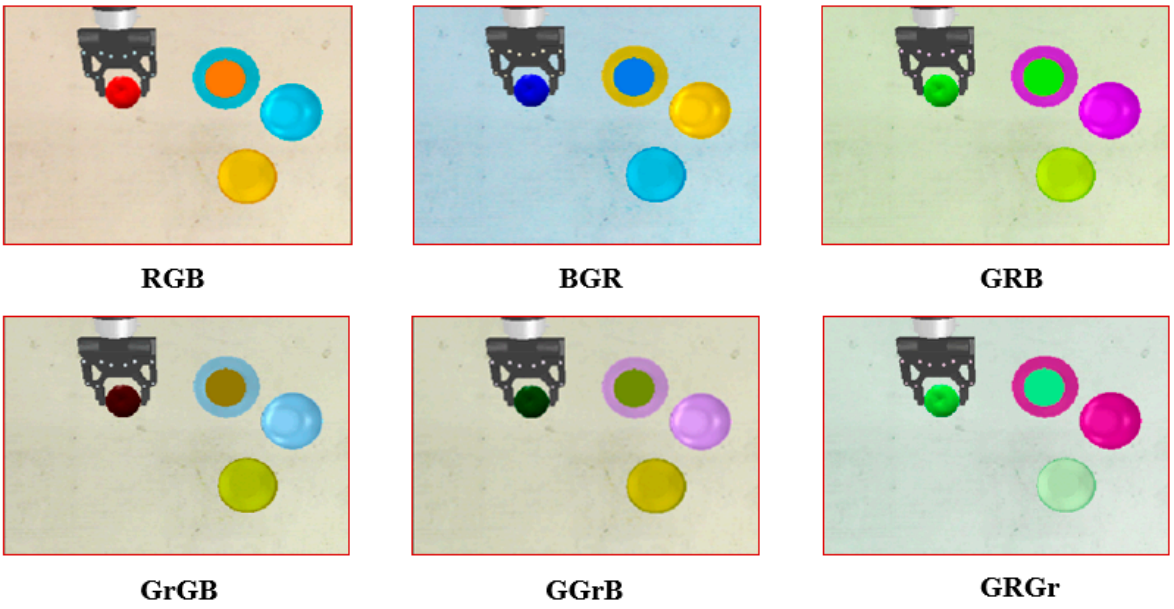

Figure 7. Augmented images with different channels.

4.3. Training the Motion Policy Network

The depth map of the environment and the pixel coordinates of the center of the target container are fed into the motion policy network and it outputs the relative position of the end-effector and the state of the gripper. The initial depth map with size of 240×320 is first cropped at the center by a window of size 224×288. To increase the robustness to the height of the camera, a disturbed value δ is added to each depth map and δ is uniformly distributed from -0.1 to 0.1. Further, a Gaussian noise with mean value of 0 and standard deviation of 0.003 is also added to each pixel value in the depth map for reducing the sim-to-real gap [34]. In addition, a stochastic disturbance within six pixels is added to the pixel coordinates of the center of the target container. The state of the gripper only

changes at the end of each trajectory. Therefore, the positive samples and the negative samples are highly unbalanced and the training sample at the end of each trajectory is duplicated for three times to reduce this imbalance [35]. A dropout layer with dropout rate of 0.3 is applied between each dense layer. Parameters of Adam optimizer and related parameters for network training are provided in Table 1. After training, the motion policy network can generate a motion action based on the depth map of the environment and the semantic keypoint.

4.4. Testing with Different Kernels and Distractors

The aim of this placing task is to place an object into the target container in the presence of two distractors. Twenty novel containers and twenty new objects are chosen in testing experiments, shown in Figure 8. For testing the generalization ability, the camera is moved in height direction with a short distance, change the relative position of the robot and tune the illumination. As a result, the new testing environment is shown in Figure 9. Two kernels are integrated in the object detection module and the influence of different kernels is explored. The object detection network is trained with different kernel independently with the same parameters as the integrated framework. The placing task are executed for 100 times in the trained scene and the new scene with three different object detection networks, and the result can be found in Table 2. It is found out that the manipulation success rate is mainly based on the precision of detection network and integrating different kernels can obtain the best performance with success rate of 99% and 95% in the trained scene and the new scene respectively. The framework with small kernel size has success rate of 92% and 90% in these two scenes. It means this framework has a good generalization ability while the original performance is relatively weak. The framework with big kernel size has a good performance in the trained scene while its performance significantly weakens in the new scene, which obtains success rate of 98% and 92% respectively. Therefore, these two kernels are integrated to get a good performance and obtain a competitive generalization ability.

Table 2. The result with different kernels in the object detection network.

Kernel Size	Detection Precision		Success Rate		Reduction Rate
	Trained Scene	New Scene	Trained Scene	New Scene	
3×3	92%	90%	92%	90%	2.17%
5×5	99%	92%	98%	92%	6.12%
Integration	100%	95%	99%	95%	4.04%

Figure 8. Novel objects and containers in the testing phase.

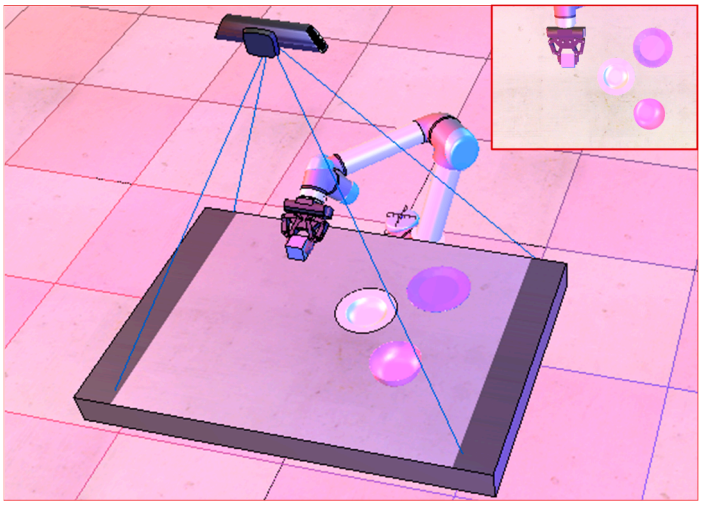

Figure 9. New environment in the testing phase.

The performance of the proposed method in the crowded environment is also explored. Another seven objects are also placed on the table randomly, which is shown in Figure 10. The placing task are executed for 100 times in the trained scene and the new scene with multiple objects based on the proposed framework, and the results are provided in Table 3. Simple scene means the scene with three containers while crowded scene is the scene with multiple objects that is shown in Figure 10. Successful and failed examples of the detection network in different scenes are provided in Figure 11. The result validates that both the object detection network and the motion policy network are efficient in crowded scene. In the crowded trained scene, the proposed method still obtains a high success rate of 98%. These experiments validate that the proposed framework can finish the object placing task efficiently for novel containers in new environment with only a demonstration and different kernels have different characteristics. They are the answers to Question 1 and Question 2. Next, the author tries to answer the Question 3.

Table 3. The result of the proposed method in different scenes.

	Detection Precision		**Success Rate**	
	simple	crowded	simple	crowded
Trained Scene	100%	99%	99%	98%
New Scene	95%	95%	94%	93%

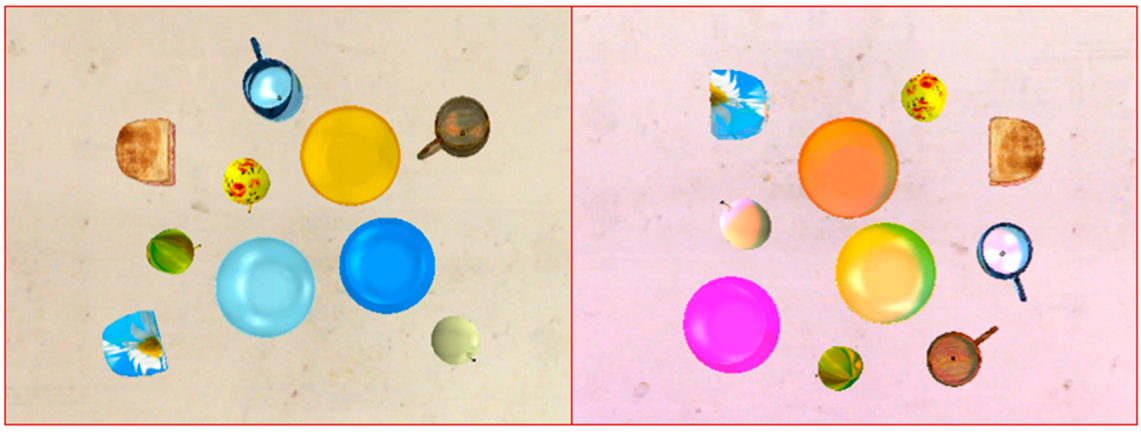

Figure 10. Different scenes with multiple objects placed on the table.

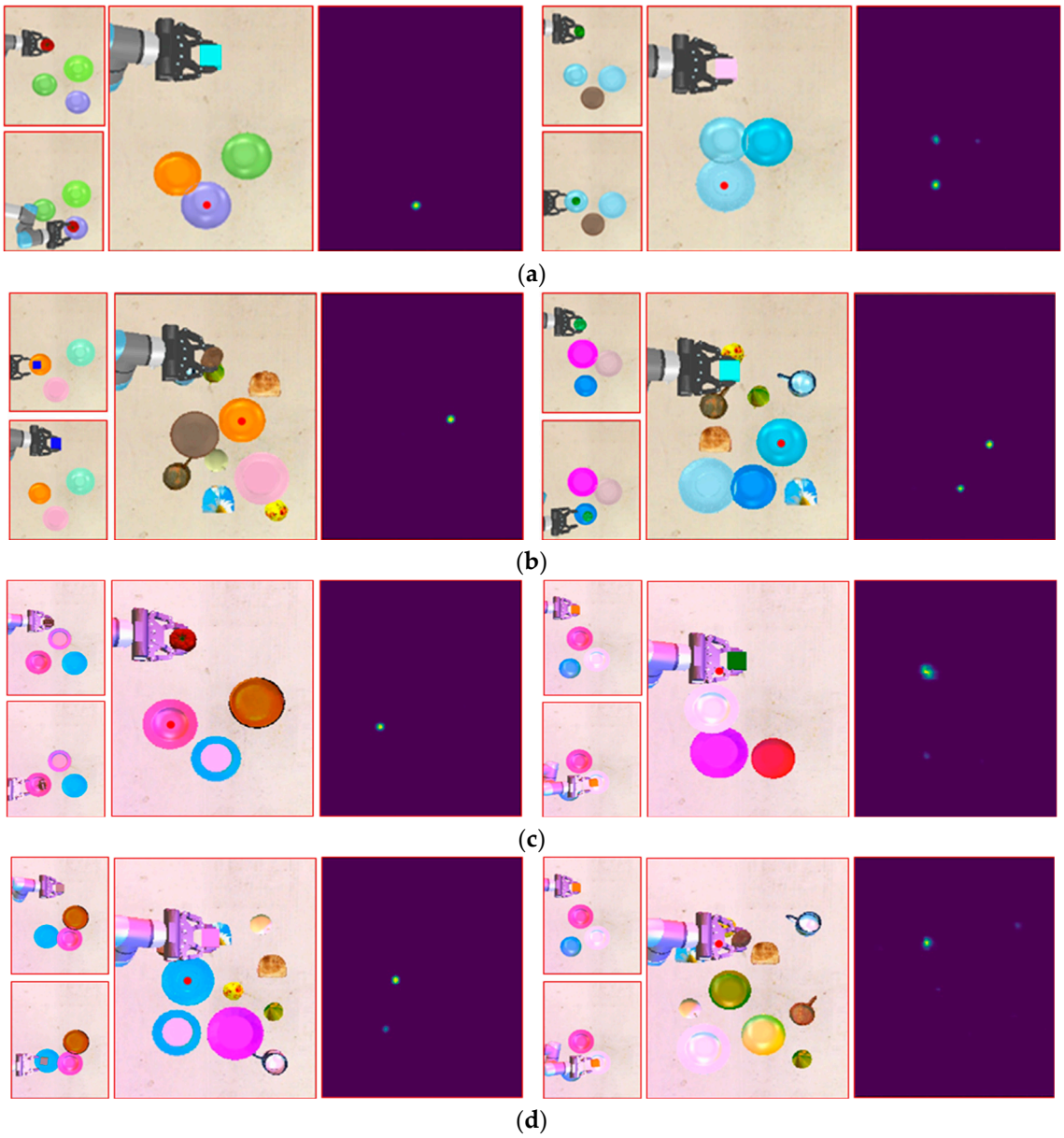

Figure 11. Examples of the detection network in different scenes. (**a**) Examples of detection network in trained scene with two distractors; (**b**) Examples of detection network in trained scene with multiple objects; (**c**) Examples of detection network in new scene with two distractors; (**d**) Examples of detection network in new scene with multiple objects.

4.5. Sim-to-Real Experiment and Comparation

In this subsection, the proposed framework is transferred from simulation to the real world directly without any fine-tuning. The experimental platform includes a Realsense D435 RGB-D camera, a UR5 robot equipped with a Robotiq gripper, seven containers and seven objects, as shown in Figures 12 and 13. The RGB-D camera outputs an image and a depth map of size 480×640. These images are resized into images of 240×320, which is the size of image in simulation. To match the input size of the networks, they are cropped by a window of size 224×288 at the center of the image and depth map. For testing, three containers are placed on the table and the robot tries to place an object to the target container in the presence of two distractors. This placing task is executed for 100 times and the result can be found in Table 4. Without any fine-tuning, the trained framework in simulation obtains a success rate of 85% in the real world. It has a better performance compared with MIL [4]

and TecNets [5]. Semantic understanding is coupled with motion policy in both MIL and TecNets framework. This mechanism makes the learned policy to be sensitive to distractors in the visual data and weakens the generalization ability. The proposed framework separates the semantic understanding from motion control module, which has a stronger robustness in different environments. In addition, without the color and texture information, the depth map has a good performance in the sim-to-real situation. With these real experiments, it can be found out that the proposed framework has a good generalization ability and robustness. They also give the answer to Question 3.

Figure 12. Containers and objects in the real world.

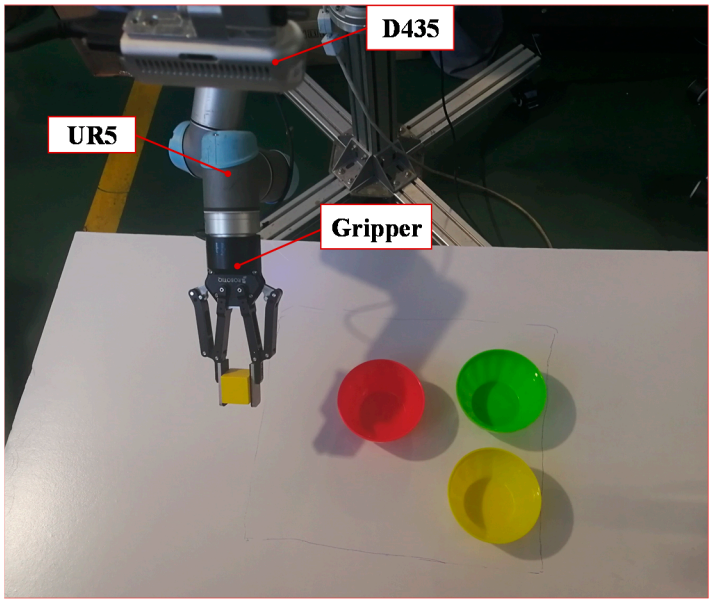

Figure 13. The experimental platform in the real world.

Table 4. The result of different methods.

Methods	Types	Success Rate
MIL(vision) [4]	Real	68.33%
TecNets(vision) [5]	Sim-to-real	72.97%
The Proposed (real world)	Sim-to-real	85%
The Proposed (trained scene)	Simulation	99%
The Proposed (new scene)	Sim-to-sim	95%

5. Conclusions and Future Work

This paper proposes an object detection-based one-shot imitation learning framework. It enables a robot to acquire similar manipulation skills efficiently and to have the ability to cope with novel objects in different environments with a single demonstration. This approach separates semantic understanding from motion control and semantic keypoint representation makes the manipulation skill more robust to different situations. Different simulated and real experimental results demonstrate the effectiveness of this proposed method. For future work, the point clouds information is considered to be applied in this framework and this framework would also be extended to more difficult manipulation tasks.

Author Contributions: Methodology, Q.S., J.H.; Software, Y.F., J.M.; Formal Analysis, Q.S., J.Q.; Investigation, Q.S., W.W.; Data Curation, Q.S., Y.F.; Writing—Original Draft Preparation, Q.S.; Writing—Review and Editing, J.Q., J.H.; Simulation and Experiment, J.M., Y.F.; and Funding Acquisition, J.Q., W.W., J.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by Special Program for Innovation Method of the Ministry of Science and Technology, China (2018IM020100), National Natural Science Foundation of China (51975360, 51775332, 51675329, 51675342), National Social Science Foundation of China (17ZDA020), and the Cross Fund for medical and Engineering of Shanghai Jiao Tong University (IH2018QNB03, YG2017QN61).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Levine, S.; Finn, C.; Darrell, T.; Abbeel, P. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.* **2016**, *17*, 1334–1373.
2. Coline, D.; Abbeel, P.; Darrell, T.; Levine, S. Deep object-centric representations for generalizable robot learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018.
3. Stephen, J.; Davison, A.J.; Johns, E. Transferring End-to-End Visuomotor Control from Simulation to Real World for a Multi-Stage Task. In Proceedings of the Conference on Robot Learning, Mountain View, CA, USA, 13–15 November 2017.
4. Chelsea, F.; Yu, T.; Zhang, T.; Abbeel, P.; Levine, S. One-Shot Visual Imitation Learning via Meta-Learning. In Proceedings of the Conference on Robot Learning, Mountain View, CA, USA, 13–15 November 2017.
5. James, S.; Bloesch, M.; Davison, A.J. Task-Embedded Control Networks for Few-Shot Imitation Learning. In Proceedings of the Conference on Robot Learning, Zurich, Switzerland, 29–31 October 2018.
6. Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; Yan, J. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
7. Manuelli, L.; Gao, W.; Florence, P.; Tedrake, R. kPAM: KeyPoint Affordances for Category-Level Robotic Manipulation. *arXiv* **2019**, arXiv:1903.06684.
8. Du, G.; Wang, K.; Lian, S. Vision-based Robotic Grasping from Object Localization, Pose Estimation, Grasp Detection to Motion Planning: A Review. *arXiv* **2019**, arXiv:1905.06658.
9. Pinto, L.; Gupta, A. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016.
10. Zeng, A.; Song, S.; Welker, S.; Lee, J.; Rodriguez, A.; Funkhouser, T. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.
11. Sui, H.; Shang, W.; Li, X. Transfer of Robot Perception Module with Adversarial Learning. *IEEE Access* **2019**, *7*, 79726–79736. [CrossRef]
12. Lampe, T.; Riedmiller, M. Acquiring visual servoing reaching and grasping skills using neural reinforcement learning. In Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, TX, USA, 4–9 August 2013; pp. 1–8.

13. Sascha, L.; Riedmiller, M.; Voigtländer, A. Autonomous reinforcement learning on raw visual input data in a real world application. In Proceedings of the 2012 International Joint Conference on Neural Networks (IJCNN), Brisbane, Australia, 10–15 June 2012; pp. 1–8.
14. Sergey, L.; Wagener, N.; Abbeel, P. Learning contact-rich manipulation skills with guided policy search. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 156–163.
15. Shao, Q.; Hu, J.; Wang, W.; Fang, Y.; Han, M.; Qi, J.; Ma, J. Composable Instructions and Prospection Guided Visuomotor Control for Robotic Manipulation. *Int. J. Comput. Intell. Syst.* **2019**, *12*, 1221–1231. [CrossRef]
16. Lee, M.A.; Zhu, Y.; Srinivasan, K.; Shah, P.; Savarese, S.; Fei-Fei, L.; Garg, A.; Bohg, J. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019.
17. Kalashnikov, D.; Irpan, A.; Pastor, P.; Ibarz, J.; Herzog, A.; Jang, E.; Quillen, D.; Holly, E.; Mrinal Kalakrishnan, V.V.; et al. Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation. In Proceedings of the Conference on Robot Learning, Zurich, Switzerland, 29–31 October 2018.
18. Chen, X.; Ghadirzadeh, A.; Folkesson, J.; Jensfeltc, P. Deep reinforcement learning to acquire navigation skills for wheel-legged robots in complex environments. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.
19. Zhang, T.; McCarthy, Z.; Jow, O.; Lee, D.; Chen, X.; Goldberg, K.; Abbeel, P. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018.
20. Hadfield-Menell, D.; Dragan, A.; Abbeel, P.; Russell, S. Cooperative inverse reinforcement learning. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016.
21. Rahmatizadeh, R.; Abolghasemi, P.; Bölöni, L.; Levine, S. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018.
22. Abolghasemi, P.; Mazaheri, A.; Shah, M.; Bölöni, L. Pay Attention—Robustifying a Deep Visuomotor Policy Through Task-Focused Visual Attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
23. Hämäläinen, A.; Arndt, K.; Ghadirzadeh, A.; Kyrki, V. Affordance Learning for End-to-End Visuomotor Robot Control. *arXiv* **2019**, arXiv:1903.04053.
24. Chen, X.; Ghadirzadeh, A.; Björkman, M.; Jensfelt, P. Adversarial Feature Training for Generalizable Robotic Visuomotor Control. *arXiv* **2019**, arXiv:1909.07745.
25. Wang, C.; Martín-Martín, R.; Xu, D.; Lv, J.; Lu, C.; Fei-Fei, L.; Savarese, S.; Zhu, Y. 6-PACK: Category-level 6D Pose Tracker with Anchor-Based Keypoints. *arXiv* **2019**, arXiv:1910.10750.
26. Bertinetto, L.; Valmadre, J.; Henriques, J.F.; Vedaldi, A.; Torr, P.H.S. Fully-convolutional siamese networks for object tracking. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Cham, Switzerland, 2016.
27. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015.
28. Li, B.; Yan, J.; Wu, W.; Zhu, Z.; Hu, X. High performance visual tracking with siamese region proposal network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
29. Shaban, A.; Bansal, S.; Liu, Z.; Essa, I.; Boots, B. One-shot learning for semantic segmentation. *arXiv* **2017**, arXiv:1709.03410.
30. Rakelly, K.; Shelhamer, E.; Darrell, T.; Efros, A.A.; Levine, S. Few-shot segmentation propagation with guided networks. *arXiv* **2018**, arXiv:1806.07373.
31. Tai, L.; Zhang, J.; Liu, M.; Burgard, W. Socially compliant navigation through raw depth inputs with generative adversarial imitation learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018.
32. Chen, G.; Yu, H.; Dong, W.; Sheng, X.; Zhu, X.; Ding, H. Learning to Navigate from Simulation via Spatial and Semantic Information Synthesis. *arXiv* **2019**, arXiv:1910.05758.

33. Morrison, D.; Corke, P.; Leitner, J. Learning robust, real-time, reactive robotic grasping. *Int. J. Robot. Res.* **2019**. [CrossRef]
34. Qin, Y.; Chen, R.; Zhu, H.; Song, M.; Xu, J.; Su, H. S4G: Amodal Single-view Single-Shot SE (3) Grasp Detection in Cluttered Scenes. *arXiv* **2019**, arXiv:1910.14218.
35. David, H.; Eck, D. A neural representation of sketch drawings. *arXiv* **2017**, arXiv:1704.03477.