Domain centralization and cross-modal reinforcement learning for vision-based robotic manipulation

Kai Yang¹, Zhanpeng Zhang², Hui Cheng¹*, Huadong Wu¹, Ziying Guo¹
(1. Department of Data Science and Computer, Sun Yat-sen University, Guangzhou 510642, China; 2. SenseTime Group Limited, Shenzhen 518000, China)

Abstract: Vision-based robotic manipulation with deep learning method has achieved substantial advances in the field of automatic agriculture, which can be deployed and applied in the picking, sorting and transporting of agricultural products and so on. Deep reinforcement learning (DRL) is one of the learning-methods that help the robot learn the policy itself by exploration and exploitation. Training real robots with DRL would take a great price that limits its application scope. Some approaches train the DRL policy in simulation and deploy the model to real robot by transferring the images in simulator to that of the real world. However, this method requires pre-collected images as training data for each real scene. In this paper, a domain centralized approach is proposed as the sim-to-real perception module to capture the task-specific characteristics of the vision regardless of the reality gap between simulation and the real environments. Another challenge for vision-based manipulation is the learning difficulty caused by the high-dimensional vision input. Here we propose a cross-modal reinforcement learning scheme by leveraging the full system states to provide additional guidance. The experimental results show that the proposed method can perform a real robot grasping task without real-world data and outperforms current methods with the same experimental settings.

Keywords: sim-to-real, robotic manipulation, agricultural application, domain centralization, cross-modal learning

DOI: 10.33440/j.ijpaa.20200302.77

Citation: Yang K, Zhang Z P, Cheng H, Wu H D, Guo Z Y. Domain centralization and cross-model reinforcement learning for vision-based robotic manipulation. Int J Precis Agric Aviat, 2020; 3(2): 48–55.

1 Introduction

Robotic manipulation, such as robot arm grasping or absorbing is essential for many applications like logistics, manufacturing and automatic agriculture. Robotic manipulation technology can be used to pick, sort and transport the agricultural products (as shown in Figure 1), which greatly reduces the labor requirements. Different from robotic manipulation with low-dimensional movement representation, vision-based methods need to extract compact features from images or videos, based on which we obtain the control command (e.g., joint angles or position of end-effector) by a trained controller, which can better distinguish the types and maturity of fruits and crops, achieving better management. The learning-based method used in vision-based robotic manipulation has shown promising results in the automatic agriculture tasks [1-3].

We propose an encoder-decoder framework to extract vision features. In this framework, the input is the randomized scene and the output is the corresponding scene without randomization (i.e., the scene before randomized). The output feature of the encoder would keep the task-related characteristics and ignore the environment variations. In this way, domains with different variations can be aligned to a task-specific subspace and we call this method domain centralization (DC). Note that our method is different from domain adaptation [4-5] and we no longer need the additional data from the real world to generalize our method to another domain. Comparing with domain randomization [6,7], we let the model learn the task-specific features automatically rather than the hand-engineered features designed by human. It means that we do not need to decide explicitly which features should be kept and the task-specific features are extracted automatically during training with the DC method. It is the main difference between DR and the proposed method. Experiments show that our method is robust to the reality gap.

Figure 1 Illustration of professional service robots used in agriculture [9].

The difficulty of training an agent is increasing with the increase of the input dimension. Fortunately, the full states of the simulator environment can be beneficial to guide RL training. Pinto et al. [9] used low-dimensional observation (i.e., the full states of environment like object coordinates) critic network and high-dimensional observation (i.e., visual images) actor network to solve the problem in an actor-critic reinforcement learning.
framework. The low-dimensional input for the critic network avoids the situation that the critic learns slower than the actor and provides ineffective guidance. However, this method is not efficient enough because the policy is nearly meaningless at the beginning of the exploration process, causing the agent unable to obtain a positive reward to optimize its policy. To solve this problem, we use a cross-modal (CM) learning scheme. In particular, we train an actor (student) with high-dimensional input dynamically guided by both low-dimensional input actor and critic (teacher). Different from multimodal learning for RL[10], we only use the visual input in the inference process.

Our contributions can be concluded as follows:

- We propose a domain centralization method to combine the advantages of both domain adaptation and domain randomization that can learn the task-specific features explicitly without the real-world data. We train the policy with high-dimensional observations obtained from domain centralization with a proposed cross-modal reinforcement learning method.
- The proposed method outperforms other current methods[4,9,11] on the vision-based grasping task. Different from modularized methods proposed by Mahler et al.[12], Viereck et al.[13] and James et al.[14], our method is an end-to-end control robotic method and it can also handle viewpoint variations to some extent.

2 Related Work

Learning-based method used in vision-based robotic manipulation can be divided into two categories: the supervised learning and the reinforcement learning method. For supervised learning methods, some algorithms let the robot collect and label data in the real world directly. The problem is that collecting the labeled data used for supervised learning would take a great price. Pinto and Gupta[15] took 50K tries and 700 hours on grasping in the real world to build the supervised training dataset. Levine et al.[16] collected over 800,000 grasp attempts in two months with 14 robotic manipulators. Unlike supervised learning, reinforcement learning (RL) does not need the labeled data. The model is trained based on the reward from exploring the environment. Combining with the deep neural network, deep reinforcement learning (DRL) has been used to solve more complex tasks. In particular, the policy gradient methods[17] using actor-critic framework are effective to solve continuous control problems. However, the potential physical damage to the robots from interacting with the environment would make the experiments unsustainable and parlous. The robots will also suffer the wear and tear while continuously operating in the data collecting and training process. When using DRL in the robotic domain, the price is higher than that of supervised learning because the policy is almost impossible to produce the correct actions and obtain the positive reward at the beginning, and the wrong actions would lead to unpredictable results. Kalashnikov et al.[18] has used 580K real-world grasping attempts to train a DQN network to perform grasping tasks. It shows impressive accuracy but the whole training process costs 4 months and over 800 robot hours with 7 robots.

Training the policy in the simulator and deploying it to the real world is an effective way to reduce the cost of policy training. However, the gap between the simulator and the real world would make the policy trained in simulation unable to expand to the real environment directly. It is due to the reality gap between the simulation and reality (e.g., the complex reflection and refraction of the light), which would reduce the generalization performance of the policy. Currently, there are two general methods to bridge the visual gap between simulation and reality. One is to transfer the observation from the simulation domain to the real domain. Bousmalis et al.[19] proposed a domain adaptation method to transfer the simulation visual images to the real images with 939,777 labeled real-world samples. Bousmalis et al.[19] learned to extract image representations both in source and target domains but only source domain contains ground truth. Wulfmeier et al.[20] used the robot’s state sequence to train a discriminator to classify the simulation-robot of fixed length state sequences. The disadvantage of the domain adaptation method is the high cost during collecting large amounts of samples in the real world.

The other method is to use domain randomization to ignore the environment variations and focus on the task-specific factors. Domain randomization adds variations on the environment during training, making the network robust to the varieties of the irrelevant variations (e.g., background texture[6] and light). Tobin et al. used domain randomization to predict the position of the target object in the scenes, then used the prediction result as observation to train the agent and deploy it to real. Sadeghi et al.[21] showed how to learn viewpoint-invariant visual servoing skills with domain randomization in a robotic manipulation scenario. However, the domain randomization needs the hand-engineered feature as the label like the object position used in[6]. Such a low-dimensional feature may lose the potential information in the scenes. James et al.[14] proposed RCAN network and transformed an input image into a simulated but not randomized image. They trained an RL policy using the transformed image as input. However, an RL policy still needs to be trained with high-dimensional input, which is challenging and requires substantial training efforts. We consider our method as a lightweight RCAN. The RCAN is trained with depth images and segmentation images as additional ground truth, which means that it is also more computationally taxing than our method. We only use the RGB images as input and ground truth, and we use the latent variable as the input of RL. The training consumption is less than the RCAN network. The reality gap existing in non-vision tasks also affects the performance of the policy. The domain adaptation and domain randomization method can also be used to solve the reality gap problem in such non-vision tasks[22].

3 Background and Preliminaries

We choose DDPG as our base RL model to obtain the policy, which is built with an actor network \( \mu(s; \theta^\mu) \) and a critic network \( Q(s, a; \theta^Q) \), while the actor network obtains actions from observations and the critic network evaluates the actor’s performance.

The critic is updated by the loss function formulated as:

\[
L = \frac{1}{N} \sum_{i} [r_i + \gamma Q(s_{i+1}, \mu(s_{i+1})) - Q(s_i, a_i)]^2
\]  

(1)

And the actor is updated with the gradient guided by critic:

\[
\nabla_{\theta^\mu} \mu \approx \frac{1}{N} \sum_{i} \nabla_a Q(s_i, a_i) \big|_{s_i, \mu(s_i)} \nabla_{\theta^\mu} \mu (s_i)
\]

(2)

where, \( N \) is the batch size and \( \mu(s; \theta^\mu) \) is the target actor network as well as \( Q(s, a; \theta^Q) \) is the target Q network referred in[22].

4 Method

In this section, we will introduce our method with two parts. In Sec 1, we introduce the domain centralization method, which aligns the input image to a task-specific subspace to alleviate the
reality gap. In Sec. 2, we introduce how to use the cross-modal reinforcement learning to train a student agent. We train the student agent with the task-specific characteristics and use the teacher agent with full state observations to guide its policy training.

### 4.1 Domain Centralization for Feature Extraction

To train the agent in the vision-based robotic manipulation task with reinforcement learning, we need to reduce the dimension of the vision-based observation and extract the pivotal features to represent the relevant variations. Here we propose domain centralization (DC), formulated in an encoder-decoder framework as shown in Figure 2a, to extract the task-specific feature from the visual input. We collect two types of images in the simulator: the original images and the corresponding randomized images. Randomized images are obtained by adding environment randomized variations (e.g., background texture, camera viewpoint, and light) on the original images. The corresponding original and randomized images have the same task-relevant characteristics. For the encoder network, it takes randomized images as input and extracts the task-specific feature as output. For the decoder network, it is designed to generate the images identical to the corresponding original images from the task-specific feature. DC combines the encoder and decoder framework to extract the features from visions automatically without labeling and can be generalized to the real-world environment without additional real data.

A straightforward way to implement our method is to design the loss function that encourages the output images to be identical to the original images. However, it is found that the background pixels compose most of the pixels of the images, causing the training process cannot capture the objects’ characteristics. To emphasize what we concern about, we put a mask on the loss function. Mask images come from the differences between the original images and the background image obtained from deleting all objects in the scenes.

Let \( \phi \in \mathbb{R}^l \) represent the task-specific feature with length \( l \) extracted from the encoder network \( D(\phi | \theta^d) \). \( f \) is the input randomized image. \( f' \) is the image generated from the decoder network \( D(\phi | \theta^d) \), where \( \theta^d \) and \( \theta^p \) denote the network parameters. \( M \) is the mask image obtained from the absolute differences between the original images \( f \) and the background images \( f':M = |f' - f| \). The loss function for training DC is formulated as:

\[
L_{DC} = \frac{1}{NCWH} \sum_{t \in \text{episode}} \left( \text{sign}(M_{vsk}) + 1 \right) (I_{vsk} - I_{vsk}')^2
\]

where, \( N \) is the batch size and \( C, W, H \) are the channel, width, and height of the images, respectively; \( \text{sign} \) is the Signum function; +1 is a bias to keep the gradient pass to the network because the background in the screen is still important.

The task-specific feature obtained from DC will be used as the observation for the RL process. In this way, we can reduce the time consumption for feature learning during RL training, and also reduce the consumption of buffer memory storing exploration transitions in DDPG. Besides, this self-supervised learning method is more data-efficient than using only the reward in RL. When deploying DC to real, we use only the encoder network to obtain task-specific feature from real images to provide observation for RL agent.

![Figure 2](https://example-image-url.com)

**Figure 2** Overview of the proposed method. (a) shows the DC network. The encoder network receives the randomized images as input and generates task-specific feature. The task-specific feature is sent to the decoder network to generate the corresponding original images. (b) is the CM reinforcement learning scheme. We use the task-specific feature and the current robot joint angles as the student actor’s input. Teacher agent is another agent which assists student agent to learn more efficiently. The teacher actor and critic use the full states information from the environment. The student agent learns the policy both from interacting with the environment and imitating the teacher agent.

### 4.2 Cross-Modal Reinforcement Learning

Training an agent with high-dimensional observation is difficult and time-consuming because of the exploration and exploitation problem. To solve the high-dimensional tasks, we propose a cross-modal (CM) reinforcement learning approach. Different from conventional RL, our approach contains a teacher agent and a student agent. As shown in Figure 2b, there are three networks for CM learning: the teacher actor \( \mu(s_{t} | \theta^t) \) and critic \( Q(s_{t}, a_{t} | \theta^c) \), trained with low-dimensional observation (i.e., full states observation) and student actor \( \mu(s_{t} | \theta^s) \), trained with high-dimensional observation (i.e., vision features and robot status). \( s_{H} \) and \( s_{L} \) represent the high-dimensional observation and low-dimensional observation, respectively. We first train the teacher agent to obtain the available base policy with DDPG. To imitate the teacher, behavior cloning\(^{[23]}\) will be added to train the student. We have an augmented loss function for student actor that contains two parts: exploration loss and behavior cloning loss, which is shown as follow:

\[
L(\theta^s) = \frac{1}{N} \sum_{t \in \text{episode}} \left[ -\alpha Q(s_{t}, a_{t} | \theta^c) \right]_{\theta^c(t+1)} + \beta \left[ \mu(s_{t} | \theta^t) - \mu(s_{H} | \theta^t) \right]_{\theta^s(t)}^2
\]

The \( \alpha \) and \( \beta \) are the factors that weight the two losses. We design the factor \( \alpha = 1 \) and \( \beta = 5e^{-\frac{\text{episode}}{5000}} \), where \#episode is
the current episode counter. Intuitively, \( \beta \) decreases gradually to let the agent learn more from the exploration rather than imitating the teacher with an increase of episodes. The reason is that the guidance from exploration experiences is more environment-specific than the guidance from behavior cloning in performance improvement since the learning process of the student is different from that of the teacher (e.g., the student learning process may encounter unseen situations to the teacher). The first part of Eqn. (4) would make the \( \mu(s_t|\theta_t) \) prefer to generate the actions that make the \( Q(s_t, a|\theta) \) have a higher value. And the behavior cloning makes the \( \mu(s_t|\theta) \) behave like \( \mu(s_t|\theta_t) \) with given states \( s_t \) and \( s_t \) from the same scene. Both student and teacher will be updated during training the student agent. It is reasonable that both the teacher actor and critic can benefit from the fine-tuning with the training samples from student’s exploration, which makes the guidance from the teacher more pertinent to the current training process. Since the teacher agent is pre-trained, we update it with a lower learning rate than that of the student agent.

The advantages of CM are: (1) Behavior cloning offers a way for the student actor to imitate the actions of the teacher actor. We can consider such a process as supervised learning because the actions from the teacher actor will be the ground truth to guide the student actor, allowing the student to learn an available model quickly at the beginning of the training process. (2) In addition to the behavior cloning method, the experiences from exploration provide another way to find the optimal policy during training. The policy gradient from the transition samples will make the student agent learn how to accomplish the tasks according to the current training process. (3) Using the pre-trained teacher’s critic to guide student actor avoids the error updating at the beginning of the training process. And the low-dimensional input critic is more efficient to guide the student actor. (4) Training teacher and student agent with different inputs and separating to two networks make the student agent can be trained in simulation but used in real world. We only provide vision input for the agent to solve the tasks.

The full learning process is shown in Algo. 1:

Algorithm 1  DC and CM for training student agent in simulation

1: Pre-train the teacher actor \( \mu(s_t|\theta_t) \) and critic \( Q(s_t, a|\theta) \) with DDPG\(^{[22]} \).
2: Pre-train DC mode with simulation data as in Sec. 1.
3: while epoch less than the maximum epochs do
4:    Environment and task initialization.
5:    Randomize environment variations, and obtain image \( \bar{f} \).
6:    Generate task-specific feature \( \phi \) from encoder \( E(\bar{f}|\theta) \).
7:    for each rolling step \( t \) do
8:        Obtain joint states \( \bar{s} \) and \( s_t = \{\phi, \bar{f}\} \).
9:        Obtain low-dimensional observation \( s_t \).
10:       Obtain actions \( a_t \) from \( \mu(s_t | \theta) \).
11:       Take \( a_t \) in simulation environments.
12:       Obtain and store \( (s_{t-1}, s_t, a_t, r_t, s_{t+1}) \) into buffer \( B \).
13:    end for
14:    for each training step \( b \) do
15:        Take a batch of samples from buffer \( B \).
16:        Update \( \mu(s_t|\theta_t) \) with Eqn. (4).
17:        Update \( \mu(s_t|\theta_t) \) and \( Q(s_t, a|\theta) \).
18:    end for
19:  end while

5  Experiment

5.1  Hardware Setup and Implementation Details

We design a robotic grasping task similar to\(^{[6,9,24–26]} \) but with more configuration variations to evaluate the DC and CM approach. We use V-REP\(^{[27]} \) as the simulator to train and evaluate the RL agent. The hardware setup is illustrated in Figure 3. The UR5 robot is a 6-DoF manipulator and we use Robotiq140 as the end-effector to interact with the objects. The actions for both the teacher actor and the student actor are the joint angles to move with the range from \(-5^\circ \) to \(5^\circ \) in each step. There are some distractors and one target object (as shown in Figure 3) randomly placed in a 40 cm\(^2\) area on the table. The target object is chosen by setting the reward. When the agent grasps the correct target object, the reward will be positive. If we want to change another target to be grasped, the RL algorithm should be retrained. However, DC can still be used because the task-specific feature will record all the objects’ information. Grasping a predefined object amongst distractors with joint angle control and sparse reward is still challenging because the least prior knowledge is available to the agent. In the previous methods, more or less prior knowledge is used to help training the agent (e.g. when controlling the end-effector’s pose, it is assumed that the agent knows the robotic arm’s construction.) Previous methods mainly specialized on the grasping task and require prior information such as hand-eye calibration, forward and inverse kinematics, or objects’ CAD models. The maximum step number in an episode is 12, and once the current step reaches the upper limit, the end-effector would close and all the joints go back to the original position, representing the end of this episode. To prevent the gripper from touching the desktop, we use the Denavit-Hartenberg (DH) matrix of UR5 to calculate the height of the end-effector according to the forward kinematics. The episode will be stopped if the height of the gripper is lower than the minimum safety height. We design the reward function as shown in Eqn. (5) to train the agents:

\[
 r_t = \begin{cases} 
 5, & \text{if succeed} \\
 -\|p_{oe} - p_{ob}\|^2 - \|a_{oe} - a_{ob}\|^2, & \text{otherwise} 
\end{cases} 
\]

where, \( t \) is the current step in an episode and \( r_t \) is the reward obtained in step \( t \), \( p_{oe} \) and \( a_{oe} \) are the position and orientation of end-effector, respectively; \( p_{ob} \) and \( a_{ob} \) are the position and orientation of the target object, respectively.

Figure 3  Top two images shows the simulation and the real environment and the bottom two show the target object used for grasping. Note that the target object can be chosen during training the RL policy according to the applications without retraining the DC network.
We collect 54,843 original images as shown in Figure 4 and corresponding randomized images in V-REP to train the DC network. The randomization variations added on the environment include the viewpoint of the camera (±2 cm and ±5°), light and the texture of background (300 different textures). In the RL training process, we train 160 epochs (50 episodes per epoch) to obtain the policy for each agent. We pre-train the teacher by Adam optimizer with learning rate $10^{-4}$ for both actor and critic. In the process of training the student, we set the learning rate $10^{-4}$ for the student actor and $10^{-6}$ for the teacher actor and critic. Although the DC network is designed to extract the task-specific feature, the feature value may still be affected by the domain factors due to the imperfect optimization process. To reduce this effect, we still randomize the simulation scenes when training the RL policy.

Figure 4  Illustration of the randomized images with different variations in the training dataset.

5.2 Network Architecture
The domain centralization network is built with an encoder-decoder framework. We use VGG-16 as the encoder with the input image size of 128×128×3. A fully connected layer is connected to the encoder to generate the task-specific feature with length 64. We use deconvolution layers and unpooling layers with the same depth as the encoder to build the decoder network. As shown in Figure 5, the input of the teacher actor consists of a 3×1 vector including coordinates $x$, $y$ and the horizontal rotation $\theta$ of the target object, as well as the 6×1 joint angle. The input vector for the teacher critic includes the same information for the teacher actor and the actions from the policy.

Figure 5  Illustration of the network architecture of the teacher and the student agents. “fc–256+relu” denotes fully connected layer with 256 hidden units and ReLU activation function.

- DC+Behavior Cloning: In this case, we still use DC to extract the features and use CM to provide guidance. The difference is that we set the factor $\alpha$ in Eqn. (4) to 0, making the student update with only imitating from the teacher and without the guidance from RL exploration. We wonder whether the agent can learn the policy just from pure behavior cloning.

5.3 Ablation Studies
To verify the effectiveness of the proposed components, we design the following experiments both in the simulator and the real world:

- Remove DC: To compare DC+CM with DR+CM, we use a plain auto-encoder network to take the place of DC framework. The auto-encoder network is trained with the images in simulation as both input and ground truth to extract the various features. The network structure is the same as that of DC. We use CM to train the agent.
- Remove CM: We use DC to extract the task-specific feature but we train the agent with pure DDPG. Both the actor and critic network use joint angles and task-specific features as input.

| Environment distractors | Simulation | Real |
|-------------------------|------------|------|
|                         | $\sqrt{}$  | $\times$ | $\sqrt{}$  | $\times$ |
| remove CM               | 14.74      | 18.52  | 12.04      | 16.67   |
| remove DC               | 19.44      | 23.15  | 15.74      | 32.41   |
| DC+BC                   | 66.67      | 74.07  | 50.93      | 74.07   |
| DC+CM (the proposed)    | **81.23**  | **83.33** | **74.07** | **79.63** |

From Table 1 we can see that distractors will increase the difficulties of grasping task. Removing not only DC but also CM will greatly reduce the performance of the policy. When replacing DC with auto-encoder in the “remove DC” method, the
agent still cannot obtain accurate task-specific features from images, causing worse performance than “DC+CM”. In addition, the CM assists the agent training with high-dimensional input well. Because of the dimension of task-specific feature extracted from images is still too high to directly train the agent (The length of task-specific feature is well-considered since we have test different length of feature and found that 64 is the most suitable value), “DC+BC” shows that behavior cloning can be deployed to assist training. However, in the tasks with distractors, “DC+BC” obtains lower performance than “DC+CM”. This is because the teacher does not know the distractors’ information, and the student just imitating the teacher will not adapt to such tasks. It shows that the experiences from exploration are also important.

5.4 Comparison with Baseline Methods

We further compare our work with other recent methods for evaluation. The methods are shown as follows:

• Tobin et al.\[9\]: DDGP: Tobin et al. proposed to predict the position of the target object in the real world by training the policy with domain randomization. And the RL is trained with the predicting position. We implement the position prediction network $a_0(\cdot)$ with our own dataset and train a general DDPG network by using the prediction as input.

• Sim2real[11]: Sadeghi et al. proposed to learn the servoing skills automatically under severe variation in viewpoint. Here we implement the network architecture in[11] but changing the output from the end-effector movement command in Cartesian space to the joint angles as ours. As for the controller policy, Sadeghi et al. used trajectories from demonstrations to guide the training process. We use the teacher agent to collect some demonstrations to assist policy learning.

• DC+AAC[9]: Similar to our method, Pinto et al.\[9\]used the high-dimensional observation actor and full state observation critic to obtain the RL policy. Comparing with our method, they did not use the full state observation actor to provide guidance. To deploy the method to real-world experiments, we add DC to their method to reduce training consumption and bridge the gap between simulation and reality.

• Auto-Encoder+CM+Style Transfer[30]: To alleviate the domain difference, another option is to transfer the real-world image to the simulation one. Here a plain autoencoder is trained with simulation images to extract the images feature for RL training. And we transfer the real images to simulation images as the input of the auto-encoder in evaluation by the recent style transfer method[30]. We do not use the randomized image for the AE since the style transfer only uses one reference style image.

We reimplement and evaluate these methods with 4 grasping conditions in our real environment setting and set up the experiments to evaluate the performance of different methods with (a) grasping a single object without distractor; (b) grasping the target object from other 5 disturbing objects (c) covering the desktop with colorful paper to change its texture and grasping the target object from distractors; (d) changing the grasping target to an unseen irregular object with similar color. The experiment procedure is the same real-world experiment in 3 and we have 108 trials for each task. The results are shown in Table 2.

| Methods | Tasks | Origin | With distractors | Texture changed | Shape changed |
|---------|-------|--------|------------------|-----------------|--------------|
| AE+CM+ST[30] | 5.56 | 6.48 | 5.56 | 0.93 |
| DC+AAC[9] | 32.41 | 31.48 | 27.80 | 26.85 |
| Tobin et al.[9] +DDPG | 40.74 | 38.89 | 32.41 | 34.26 |
| Sim2real[11] | 63.89 | 53.70 | 37.04 | 37.96 |
| DC+CM (the proposed) | 79.63 | 74.07 | 50.93 | 66.67 |

6 Conclusion

In this work, we propose a domain centralization and cross-modal reinforcement learning approach to address the end-to-end vision-based robotic manipulation task. Camera calibration or object detection is not needed in our method and all the training processes are based on simulation without additional real-world data. The full states of the environment are used to assist the vision-based training process. The performance of our approach shows that domain centralization with cross-modal reinforcement learning scheme can provide an effective learning-based way to solve the vision-based robotic manipulation tasks.

APPENDIX

Training Process

Figure 6 shows the success rate for different conditions on grasping during training the agent in simulation. We add the teacher’s training process from initial state with randomized network parameters to show the comparison of the training process. Our proposed method obtains better performance than the others except the teacher agent. However, we should note that the teacher use low-dimensional information as input. Figure 6 shows that “DC+CM” can train the agent with higher efficiency and quality.

Figure 7 shows the performances of each method in simulation with distractors during training. Our method outperforms the other methods with the best performance of nearly 70% success rate. Note that we remove the style transfer part from “AE+CM+ST” because this is in simulation scene. “AE+CM” obtains close performance as ours due to the simple training conditions (i.e., observation randomization is removed in “AE+CM”).
Domain centralization is using the pixel-level supervised clue to construct, restore and retain the task-related information. We can obtain the image \( G_0 \) from vision sensor with environment statement \( \tilde{\nu}_0 \). After setting the environment variations as \( \tilde{\nu}_{\text{obj}} \), we obtain a new image \( G' \). In the end, we obtain \( G \) while adding the task-agnostic information \( \tilde{\nu}_{\text{noise}} \). The whole process is shown in Eq. (6):

\[
G_0 = g(\tilde{\nu}_0) = g(0) \\
G' = g(\tilde{\nu}_0 + \tilde{\nu}_{\text{obj}}) \\
G = g(\tilde{\nu}_0 + \tilde{\nu}_{\text{obj}} + \tilde{\nu}_{\text{noise}}) \tag{6}
\]

While \( g(\cdot) \) is the process of collecting the image in the simulation scenes with vision sensor.

[References]

[1] Rodríguez F, Moreno J C, Sánchez J A, and Berenguel M. Grasping in Agriculture: State-of-the-Art and Main Characteristics. Mechanisms and Machine Science, chapter 15, 2013. doi: 10.1007/978-1-4471-4664-3_15.

[2] Kentaro K, Masahiro K, and Tetsuo S. Development of the stem grasping and bioinformation measuring robot for the precision agriculture of the tomato. Hokurikushinetsu Branch, 2018.

[3] Fountas S, Borja E G, Kasimati A, et al. The Future of Digital Agriculture: Technologies and Opportunities. IT Professional, 2020, 22(1): 24–28. doi: 10.1109/MITP.2019.2963412.

[4] Bousmalis K, Irpan A, Wohlhart P, Bai Y F, Kelcey M, Kalakrishnan M, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In 2018 IEEE International Conference on Robotics and Automation, 4243–4250, 2018. doi: 10.1109/ICRA.2018.8460875.

[5] Spinello L and Arras K O. Leveraging rgb-d data: Adaptive fusion and domain adaptation for object detection. In 2012 IEEE International Conference on Robotics and Automation, pages 4469–4474. Citeseer, 2012. doi: 10.1109/ICRA.2012.6225157.

[6] Tobin J, Fong R, Ray A, Schneider J, Zaremba W, and Abbeel P. Domain randomization for transferring deep neural networks from simulation to the real world. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 23–30, 2017. doi: 10.1109/IROS.2017.8202133.

[7] Tobin J, Biewald L, Duan R, et al. Domain Randomization and Generative Models for Robotic Grasping. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018. doi: 10.1109/IROS.2018.8593933.

[8] Robotics Online Marketing Team. Agricultural robots: Understanding professional service robots in agriculture, 2019. Available: https://www.robotics.org/blogarticle.cfm/Agricultural-Robots-UnderstandingProfessional-Service-Robots-in-Agriculture/151.

[9] Pinto L, Andrychowicz M, Welinder P, Zaremba W, and Abbeel P. Asymmetric actor critic for image-based robot learning. Robotic: Science and Systems, 2018. doi: 10.15607/RSV.2018.XIV.008.

[10] Wang Y, Patel A, Shen Y L, and Jin H X. A deep reinforcement learning based multimodal coaching model (dcm) for slot filling in spoken language understanding (slu). Proc. Interspeech 2018, pages 3444–3448, 2018. doi: 10.21437/Interspeech.2018-1379.

[11] Sadeghi F, Toshev A, Jang E, and Levine S. Sim2real viewpoint invariant visual servoing by recurrent control. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4691–4699, 2018. doi: 10.1109/CVPR.2018.00493.

[12] Mahler J, Liang J, Niyaz S, Laskey M, Duan R, Liu Y X, et al. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In Robotics: Science and Systems, 2017. doi: 10.15607/RSS.2017.XIII.058.

[13] Vierецk U, Pas A T, Saenko K, and Platt R. Learning a visuomotor controller for real world robotic grasping using simulated depth images. In Proceedings of the Robot Learning, pages 291–300, 2017.

[14] James S, Wohlhart P, Kalakrishnan M, Kalashnikov D, Irpan A, Ibarz J, et al. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 12627–12637, 2019. doi: 10.1109/CVPR.2019.01291.

[15] Pinto L and Gupta A. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In 2016 IEEE International Conference on Robotics and Automation, pages 3406–3413, 2016. doi: 10.1109/IROS.2016.7747517.

[16] Levine S, Pastor P, Krizhevsky A, Ibarz J, and Quillen D. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. The International Journal of Robotics Research, 37(5-6):421–436, 2018. doi: 10.1177/0736899418795115.

[17] Schulman J, Wolski F, Dhariwal P, Radford A, and Klimov O. Proximal
policy optimization algorithms. arXiv:1707.06347, 2017.
[18] Kalashnikov D, Irpan A, Pastor P, Ibarz J, Herzog A, Jiang E, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. arXiv:1806.10293, 2018.
[19] Bousmalis K, Trigeorgis G, Silberman N, Krishnan D, and Erhan D. Domain separation networks. In Advances in neural information processing systems, pages 343–351, 2016.
[20] Wulfmeier M, Posner I, and Abbeel P. Mutual alignment transfer learning. arXiv:1707.07907, 2017.
[21] Chebotar Y, Handa A, Makoviychuk V, Macklin M, Issac J, Ratliff N, et al. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In 2019 International Conference on Robotics and Automation, pages 8973–8979. IEEE, 2019. doi: 10.1109/ICRA.2019.8793789.
[22] Lillicrap T P, Hunt J J, Pritzel A, Heess N M O, Erez T, Tassa Y, et al. Continuous control with deep reinforcement learning. US Patent App. 15/217,758. doi: 10.1016/S1098-3015(10)67722-4.
[23] Bojarski M, Testa D D, Dworakowski D, Firner B, Flepp B, Goyal P, et al. End to end learning for self-driving cars. arXiv:1604.07316, 2016.
[24] Andrychowicz M, Wolski F, Ray A, Schneider J, Fong R, Welinder P, et al. Hindsight experience replay. In Advances in Neural Information Processing Systems, pages 5048–5058, 2017.
[25] Quillen D, Jang E, Nachum O, Finn C, Ibarz J, and Levine S. Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods. In 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018. doi: 10.1109/ICRA.2018.8461039.
[26] Zhu Y K, Wang Z Y, Merel J, Rusu A A, Erez T, Cabi S, et al. Reinforcement and imitation learning for diverse visuomotor skills. Robotics: Science and Systems XIV, abs/1802.09564, 2018. doi: 10.15607/RSS.2018.XIV.009.
[27] Rohmer E, Singh S, and Freese M. V-rep: A versatile and scalable robot simulation framework. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1321–1326, 2013. doi: 10.1109/IROS.2013.6696520.
[28] Kingma D P and Ba J. Adam: A Method for Stochastic Optimization[J]. Computer Science, 2014.
[29] Singh A, Yang L, and Levine S. Gplac: Generalizing vision-based robotic skills using weakly labeled images. In Proceedings of the IEEE International Conference on Computer Vision, pages 5851–5860, 2017. doi: 10.1109/ICCV.2017.623.
[30] Johnson J, Alahi A, and Li F F. Perceptual losses for real-time style transfer and super resolution. In European conference on computer vision, pages 694–711. Springer, 2016. doi: 10.1007/978-3-319-46475-6_43.