International Journal of Computational Intelligence Systems
Vol. **12(2)**, 2019, *pp.* 1221–1231
DOI: https://doi.org/10.2991/ijcis.d.191017.001; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

Composable Instructions and Prospection Guided Visuomotor Control for Robotic Manipulation

Quanquan Shao, Jie Hu*, Weiming Wang, Yi Fang, Mingshuo Han, Jin Qi, Jin Ma

Institute of Knowledge Based Engineering, School of Mechanical Engineering, Shanghai Jiao Tong University, No. 800 Dongchuan Road, Minhang District, Shanghai, 200240, China

ARTICLE INFO

Article History
Received 17 Jul 2019
Accepted 16 Oct 2019

Keywords
Composable instructions
Motion generation
Prospection
Imitation learning
Visuomotor control
Robotic manipulation

ABSTRACT

Deep neural network-based end-to-end visuomotor control for robotic manipulation is becoming a hot issue of robotics field recently. One-hot vector is often used for multi-task situation in this framework. However, it is inflexible using one-hot vector to describe multiple tasks and transmit intentions of humans. This paper proposes a framework by combining composable instructions with visuomotor control for multi-task problems. The framework mainly consists of two modules: variational autoencoder (VAE) networks and long short-term memory (LSTM) networks. Perception information of the environment is encoded by VAE into a small latent space. The embedded perception information and composable instructions are combined by the LSTM module to guide robotic motion based on different intentions. Prospection is also used to learn the purposes of instructions, which means not only predicting the next action but also predicting a sequence of future actions at the same time. To evaluate this framework, a series of experiments are conducted in pick-and-place application scenarios. For new tasks, the framework could obtain a success rate of 91.2%, which means it has a good generalization ability.

1. INTRODUCTION

Robotic manipulation technology has been widely explored by robotics researchers in past decades. It mainly studies how the robot could move and manipulate objects for various purposes by actions including grasping [1], pushing [2], pouring [3], opening [4], picking [5], placing [6] and so on. Robotic manipulation technology does not focus on the specific robot platforms but rather concentrates on the general related methods of the robot interaction with objects [7]. It involves, traditionally, scene understanding and segmentation, object recognition, pose estimation, action plan, motion plan, task plan and so forth. Robots need to perceive the surroundings by sensors, make decisions based on the perception information and move to execute decision results. Obviously, it is an integration of related technologies to finish specified manipulation tasks. Great successes have been achieved by decomposing manipulation tasks into different stages and solving them separately. Due to the good reliability and high availability of this roadmap, robotic systems based on this roadmap have been widely deployed in various scenarios. However, the framework with separated stages is often time-consuming and less responsive to changes of surroundings rapidly. Besides, propagation of deviations between each stage would also weaken the performance of this framework [8].

With deep neural networks making great achievements in image processing, machine vision and natural language processing, deep learning method is increasingly used in robot research. As a new pipeline for robotic manipulation, end-to-end visuomotor method has been proposed in recent years [9,10]. Compared to these traditional multi-stage frameworks, this pipline does not require scene understanding, object recognition, pose estimation, action plan and motion plan. The end-to-end visuomotor method attempts to map the perception information got by sensors into motional action space directly just like a visual servoing style [9]. The perception information is typically RGB images or both RGB images and depth maps. Depending on different application scenarios, the motion action could be an absolute value or relative changes in Cartesian coordinate or joint coordinate system. The action of visuomotor control could also be joint torque in the case of force control or contact interaction.

This end-to-end framework is more time efficient than the traditional separated method, and it could adjust its action in real time based on dynamic changes in the environment [10]. Furthermore, end-to-end frameworks are easily integrated with learning-based approaches. It has been used for robotic manipulation in different situations such as grasping [11], picking up [12] and pushing [2]. In the past few years, robotics researchers and artificial intelligence (AI) scientists are actively exploring the visuomotor control approach for robotic manipulation to expand its application field, enhance its robustness, accelerate its learning speed and so on [12,13]. Some studies also have been done through this learning-based end-to-end visuomotor control for multi-task or multi-step situations [14,15]. Previous research has validated that end-to-end visuomotor control for multi-task manipulation is more

*Corresponding author. Email: hujie@sjtu.edu.cn

sample-efficient and robust compared to methods applied to single tasks [14]. One-hot vector, which means only one item of the vector is one and others are all zeros, is often used as instruction to distinguish different purposes of the robot in the same or different surroundings. However, it is inflexible using one-hot vector to describe multiple tasks and transmit intentions of humans. Tasks described by one-hot vectors also ignore the correlation of different tasks, which is very useful for generalization in robotic manipulation.

This paper tries to explore the relation of different tasks via considering the basic units of tasks. It displaces the one-hot vector with composable instruction and combines it with end-to-end visuomotor control for robotic manipulation in multi-task situation. The proposed approach mainly consists of two modules: a variational autoencoder (VAE) and a long short-term memory (LSTM) recurrent network. Perception information of surroundings is encoded by VAE into a small latent space. The embedded perception information and composable instructions are combined by the LSTM module to guide robotic movement based on different intents. Lots of execution trajectories are recorded as labeled training data. VAE is trained with images captured during execution of robotic manipulation, and then LSTM network is trained with state-action pairs, which could be regarded as policy network. Obviously, the learning style of this approach is a pure Behavior Cloning. Most methods of Behavior Cloning train the policy network with pairs consisting of current state and next action. Nevertheless, it is found that prospection mechanism is very important to get goals of the instructions in multi-task scenarios. Prospection means not only predicting the next action but also predicting a sequence of future actions at the same time.

On the whole, the primary contributions of the proposed method are twofold: (1) Considering the basic units of tasks, one-hot vector is replaced by composable instructions in the multi-task situations. As far as I know it is the first time that composable instructions are used to describe multiple tasks for robotic manipulation in the learning-based end-to-end visuomotor framework. (2) Different from the traditional BC framework, prospection mechanism is used in the training phase of the control network. Prospection mechanism tries to predict a sequence of future actions at the same time, which allows the robot to focus on the global characteristics and makes it easier to learn the intent of the instructions.

A typical multi-object pick-and-place application scenario is chosen for validation and it is similar to tasks of table clean-up or object sorting [16] for service robots in daily life. With the wide variety of manipulation tasks, learning-based methods are important ways to give robots diverse manipulation skills. Based on the proposed framework, the robot could get different skills with good generalization ability. In addition, the framework can be also used in other multi-step application scenarios including but not limited to cooking.

2. RELATED WORK

Robotic manipulation has always played an important role in the development of robotics and automation technology. Mason *et al.* surveyed the technologies of robotic manipulation and compared them with abilities of manipulation of humans [7]. Roboticists have been working to enable robots to perform manipulation tasks as intelligent and skillful as humans. Traditionally, robotic manipulation technology is divided into several parts such as perception, decision-making, motion plan and robot control. They are orderly combined after being processed separately to accomplish different manipulation tasks. Related technologies mainly include scene understanding and segmentation [17], object detection [18] and recognition [19], pose estimation [20], camera calibration [21], action plan [22], motion plan [23] and task plan [24]. Based on this technical roadmap, robots are used successfully for grasping objects [1], tidying up objects [6], cleaning up boxes [25], pouring waters [3], opening doors and drawers [4,26], picking fruits and vegetables [5], cooking [27] and so on. Due to the dynamic changes of the surrounding environment, the diversity of objects and the complexity of manipulation tasks, it is becoming harder and harder and even impossible to pre-determine every step in different robotic manipulation tasks. Various manipulation skills are gained by robots via learning-based methods, which include learning from demonstrations (LfD) [28], behavior cloning (BC) and imitation learning (IL) [29], reinforcement learning (RL) [30] as well as the combination of them.

With the great ability of feature self-extracting and powerful cross-modal mapping capability, deep neural networks have achieved a great success in scene understanding, object detection and recognition [31,32]. Similar methods are also used in the perception and decision stages of manipulation tasks and then they are combined with traditional motion plan and control methods to achieve different robotic manipulation tasks. Lenz *et al.* applied deep neural network to detect the robotic grasping points with RGB-D sensors for picking-up tasks [11]. Pinto *et al.* proposed a self-supervised grasping detection method which could obtain training data online [33]. Zeng *et al.* used an online self-learning method to combine pushing action and grasping action together for object grasping in dense clutter [2]. Zhen *et al.* used RCNN networks for object recognition and combined it with traditional pose estimation to build a scene graph, and then these scene graphs was used as goal states for high-level task plan [34]. Xu *et al.* applied a framework of neural task programming to acquire high-level semantic plan with video demonstrations [16]. As mentioned above, they are all still multi-stage pipelines, which are time-consuming and propagating uncertainty between each stage. As an alternative framework, learning-based end-to-end visuomotor methods were proposed [9,10,35]. Different from the multi-stage pipelines, these approaches combined perception and control together and mapped the perception information into robotic control actions directly via neural networks. Visuomotor control methods for robotic manipulation are often trained using IL, RL and their combinations. Levine *et al.* integrated deep neural networks with guide policy search (GPS) that was an RL method for learning contract-rich manipulation skills [35,36]. Pinto *et al.* applied an asymmetric actor-critic framework for the robot picking-up tasks and transferred it directly from simulation to real [37]. Levine *et al.* used an online visuomotor framework for grasping cluttered objects without hand-eye calibration. However, it needed 6 to 14 robots for months to collect labeled data [12]. Combining visuomotor methods with RL is normally data-inefficient and some works have been done to improve the efficiency. Finn *et al.* proposed a deep spatial autoencoder to compress the visual images and promoted the learning speed of RL in several simple robotic manipulation tasks [38]. Gu *et al.* proposed an

asynchronous off-policy update tricks in 3D dimensions and obtained a higher efficiency with multiple robots learning at the same time [13]. James *et al.* used domain randomization for robot learning in simulation and transferred the manipulation skill from simulation to the real world [8]. A deep RL framework was used for peg-in-hole tasks by combining visual inputs, proprioceptive information and torques of end effector [39].

As a supervised learning framework, BC mainly has a high data-efficiency and could make robots obtain various manipulation skills rapidly. BC directly learns control policies from demonstrations while LfD may more emphasize learning from other's demonstrations, especially from humans. A more generalized demonstration-based learning framework is known as IL. Integrating IL with visuomotor control is also widely applied for robotic manipulation. Zhang *et al.* used virtual reality headsets with hand-tracking equipment to teleoperate the manipulator for manipulation demonstration data collection and validated the efficiency of IL in complex robotic manipulation tasks [40]. Generally, the IL framework was trained with the state-action pairs. For another way, Torabi *et al.* made a preliminary exploration about behavior cloning from observation (BCO) [41]. The framework just used the trajectories of observations or states without the information of action of each state. Similar to the objective of LfD, it has a great significance that the robot could observe others accomplishing a task to get manipulation skill, which is just like humans. There are also some articles focusing on reducing numbers of demonstrations for learning a new manipulation skill rapidly, named few-shot learning or one-shot learning. Finn *et al.* applied Meta-Learning for one-shot IL, which could efficiently adjust the parameters of the policy network based on new demonstration with few-shot learning [42]. Yu *et al.* proposed a domain-adaptive meta-learning method and make the robot learn a new pick-and-place skill from one human demonstration [43].

However, these meta-learning methods adjusted the whole parameters of the policy network for new demonstrations and would forget the previous manipulation skills. James *et al.* proposed a new few-shot learning framework via a task- embedded space [44]. This approach fixed the parameters of control network and performed new demonstration task via a task-embedded vector that was like instructions that is used to transmit intentions of humans. For multi-task situation, one-hot vector was often used as instruction to distinguish different purposes of the robot in the same or different surroundings. Rahmatizadeh *et al.* validated its efficiency and robustness of learning multi-task demonstrations at the same time by using end-to-end visuomotor framework [14]. One-hot vector was used to stand for different tasks. It was also used for motion switching in multi-step task such as cloth-folding [45]. Abolghasemi *et al.* improved the robustness of this multi-task approach via task-focused visual attention [15]. Hausman *et al.* also combined an embedding task space with RL to learn transferable robot skills [46]. Motivated by these related articles, in this paper, an end-to-end visuomotor framework with composable instruction is used for multi-task learning to get interpretable and extensible robot skills.

3. METHODOLOGY

The proposed framework is based on BC, which is a direct IL framework. There are some notations of end-to-end BC for multi-task robotic manipulation. The robot tries to learn a map from its state $\boldsymbol{s}$ or observations $\boldsymbol{o}$ and instructions $\boldsymbol{I}$ to motion action $\boldsymbol{a}$, which is called as policy $\boldsymbol{\pi}(\boldsymbol{o}, \boldsymbol{I})$. Here, policy $\boldsymbol{\pi}$ is represented by deep neural network while visual images $\boldsymbol{o}$ and instruction vector $\boldsymbol{I}$ are its inputs and the outputs are the next pose vectors $\boldsymbol{a}$. In BC learning framework, policy network is learned with the obtained demonstration sets $\boldsymbol{\Gamma}$. Demonstrations $\boldsymbol{\Gamma}$ are example trajectories of expert policies $\boldsymbol{\pi}^*$ that is assumed as optimal solutions of maps from observations $\boldsymbol{o}$ to actions $\boldsymbol{a}$. In some articles, they are also explored methods about learning from suboptimal demonstrations with the BC learning framework. However, it is not the focus in this project. Demonstrations are assumed to be optimal in the proposed method. These trajectories are composed of a series of observations $\boldsymbol{o}$ and motion actions $\boldsymbol{a}$, which could be represented as $\boldsymbol{\tau} = [(\boldsymbol{o}_1, \boldsymbol{a}_1), \ldots, (\boldsymbol{o}_T, \boldsymbol{a}_T)]$. Instructions $\boldsymbol{I}$ are also saved with each trajectory, which represent different tasks in multi-task manipulation situation. Therefore, demonstrations would be noted as $\boldsymbol{\Gamma} = \{(\boldsymbol{\tau}_1, \boldsymbol{I}_1), \ldots, (\boldsymbol{\tau}_K, \boldsymbol{I}_K)\}$. The proposed end-to-end framework is trained with these expert demonstrations. After trained, the policy network could generate motion trajectories in real time to accomplish different manipulation tasks depending on the various observations and the specified instructions. Instructions could indicate what humans want the robot to do, and also represent control interfaces that exchanges information with other high-level task plan modules. The main architecture of this end-to-end visuomotor control method for robotic manipulations is illustrated in Figure 1. As mentioned above, the proposed approach mainly includes two modules: perception network and control network. Perception network is composed of a series of convolutional Resnet blocks, which make up VAE. The VAE compresses the images got by visual sensors into a low-dimensional embedding space and then a recurrent network, as the control network, is used to guide robotic motion for specific robotic manipulation tasks by combining latent variable of perception information and composable instructions. Recurrent network could encode sequential information and make prediction based on historical time series data. Detailed descriptions of perception network and control network are followed.

3.1. Perception Network

The perception of surroundings contains hardware of the robot, target objects and other disturbing objects. Normally, perception information are images obtained by a visual sensor, which are RGB images in this case. Images are very high-dimensional, especially in

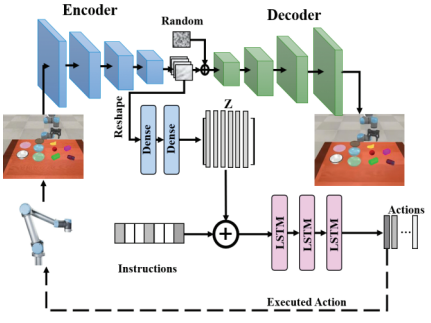


Figure 1 The framework of the proposed approach.

the case of a series of historical images and they are highly redundant for specific scenarios as well. It is very difficult to train these networks that map high-dimensional images to motion space from scratch and requires various kinds of labeled training data. Pretraining for networks that are used to process high-dimensional images information is very important, especially for networks composed of convolutional modules that could learn features for classification or regression by themselves. Compression processing is also widely used for redundant high-dimensional data such as images or video frames. Variational autoencoder [47] is often used to compress the perceptional information in field of robot vision [48–50]. It is verified that VAE has better generalization ability compared with traditional autoencoder (AE) in visuomotor framework [51]. Hoof *et al.* used VAE to process the high-dimensional tactile and visual feedback for stabilization task [48]. Hämäläinen *et al.* applied VAE to cope with the visual observation for affordance learning in a visuomotor control framework [49]. Piergiovanni compressed the image via a VAE for robot navigation [50]. Following this pipeline, a convolutional VAE module is also used for compression of the perceptual information in the proposed method, and it consists of a series of convolutional Resnet blocks [52,53]. VAE is a network that contains a bottleneck structure, which compresses high- dimensional images into a low-dimensional data. It is mainly separated into two modules: an encoder network $\boldsymbol{E}$ and a decoder network $\boldsymbol{D}$. The encoder network $\boldsymbol{E}$ maps the observations $\boldsymbol{o}$ into a latent variable $\boldsymbol{z} \sim \boldsymbol{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$. The decoder network $\boldsymbol{D}$ restores the observation $\boldsymbol{o}$ from latent variable $\boldsymbol{z}$. Compared with traditional AE, VAE also considers the neighbors of the encoded point in latent space and makes the coding process more robust. It can be formalized as:

$$\boldsymbol{VAE}: \boldsymbol{o}' = \boldsymbol{D}(\boldsymbol{z} \sim \boldsymbol{N}(\boldsymbol{\mu}, \boldsymbol{\sigma} = \boldsymbol{E}(\boldsymbol{o}))) \tag{1}$$

It assumes the restored $\boldsymbol{o}'$ equals to the input $\boldsymbol{o}$ and the observation $\boldsymbol{o}$ could be encoded as $\boldsymbol{\mu}$. Latent variable $\boldsymbol{z}$ has a much lower dimension than the observation $\boldsymbol{o}$. Normally, it also assumes that $\boldsymbol{z}$ obeys standard Gaussian distribution $\boldsymbol{z} \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I})$. The K-L divergence between the standard Gaussian distribution $\boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I})$ and the obtained distribution $\boldsymbol{N}(\boldsymbol{\mu}(\boldsymbol{o}), \boldsymbol{\sigma}(\boldsymbol{o}))$ is used to measure the difference of these distributions. In our case, the unit matrix $\boldsymbol{I}$ is assumed to be the variance of the encoded latent variable $\boldsymbol{z}$. And the Equation (1) could be written as:

$$\boldsymbol{VAE}: \boldsymbol{o}' = \boldsymbol{D}(\boldsymbol{z} \sim \boldsymbol{N}(\boldsymbol{\mu} = \boldsymbol{E}(\boldsymbol{o}), \boldsymbol{I})) \tag{2}$$

The reparameterization trick is usually utilized [47] for the non-differentiable sampling operation. It means auxiliary variables are used to replace the non-differentiable sampling operation:

$$\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\delta}, \boldsymbol{\delta} \sim N(\boldsymbol{0}, \boldsymbol{I}) \tag{3}$$

The notations of these Gaussian distribution are shown:

$$\boldsymbol{q}(\boldsymbol{z}|\boldsymbol{o}) = \boldsymbol{N}(\boldsymbol{\mu}(\boldsymbol{o}), \boldsymbol{I}) \tag{4}$$

$$\boldsymbol{p}(\boldsymbol{z}) = \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{5}$$

The loss of VAE is followed:

$$\mathcal{L}_{KL} = \boldsymbol{D}_{KL}(\boldsymbol{q}(\boldsymbol{z}|\boldsymbol{o}) \| \boldsymbol{p}(\boldsymbol{z})) \tag{6}$$

$$\mathcal{L}_{res} = \|\boldsymbol{D}(\boldsymbol{z}) - \boldsymbol{o}\|_2^2 \tag{7}$$

$$\mathcal{L}_{VAE} = \mathcal{L}_{KL} + \mathcal{L}_{res} \tag{8}$$

where $\mathcal{L}_{KL}$ is the K-L divergence between the conditional distribution $q(\boldsymbol{z}|\boldsymbol{o})$ and the normal distribution $p(\boldsymbol{z})$. $\mathcal{L}_{res}$ is the deviation of the observation $\boldsymbol{o}$ and the restored $\boldsymbol{o}'$, which is expressed with the square of Euclidean Norm and $\mathcal{L}_{VAE}$ is the total loss of VAE. Inspired by the work of Abolghasemi [15], another loss term that is the deviation of the reconstructed images of encoded vector $\boldsymbol{\mu}$ is added to enhance coding performance.

$$\mathcal{L}_{cod} = \|\boldsymbol{D}(\boldsymbol{\mu}) - \boldsymbol{o}\|_2^2 \tag{9}$$

where $\mathcal{L}_{cod}$ is the deviation of the observation $\boldsymbol{o}$ and the restored $\boldsymbol{o}'$ of $\boldsymbol{\mu}$, which is expressed with the square of Euclidean Norm. Regularization is also used to avoid overfitting, and the total loss function is modified:

$$\mathcal{L}_{VAE} = \lambda_1 \mathcal{L}_{KL} + \lambda_2 \mathcal{L}_{res} + \lambda_3 \mathcal{L}_{cod} + \lambda_4 \mathcal{L}_{reg} \tag{10}$$

where $\mathcal{L}_{reg}$ is the loss of regularization and the hyper-parameters $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are the weights of different terms in the loss function. Both encoder network and decoder network are mainly composed of Resnet blocks, and there are no fully connection layers at the end of encoder network, which is different from the traditional structure of VAE. The activation function of the last convolutional layer of encoder network is replaced by linear activation function to code the observations into a zero-center embedding space while the activation function of convolutional layer is traditionally rectified linear unit (ReLU) activation function. The two-dimensional coding framework could retain more spatial information that is very useful for robotic manipulation. The detailed structure of perception network is shown in Figure 2. The input to VAE is an RGB image of size $224 \times 288 \times 3$. The output of the convolutional encoder is a tensor of size $7 \times 9 \times 10$, which means it includes 10 feature maps of size 7×9. With the trained encoder network, the high-dimensional perception information could be embedded into a latent space of size $7 \times 9 \times 10$.

3.2. Control Network

Control network is primarily a LSTM recurrent network. It is used to guide the movement of the robot to accomplish specific manipulation task by merging embedding variable $\boldsymbol{\mu}$ of the observed image $\boldsymbol{o}$ and composable instructions $\boldsymbol{I}$. The BC learning framework is used for training this control network. It means the control network tries to make decisions similar to the optimal demonstrations $\boldsymbol{\Gamma}$. And the typical loss function of BC is followed:

$$\mathcal{L}_{bc} = \|\boldsymbol{\pi}(\boldsymbol{E}(\boldsymbol{o})_k, \boldsymbol{I}) - \boldsymbol{a}_k\|_2^2 \tag{11}$$

where $\boldsymbol{a}_k$ is optimal action of the observation $\boldsymbol{o}$ based on the specified instructions $\boldsymbol{I}$ in demonstrations $\boldsymbol{\Gamma}$. However, this sequential decision-making problem of robotic manipulation is not ideally Markov decision process (MDP), and the action depends not only on the current state or observation but also on the historical trajectory of state or observation, especially in complex situations [14,15].

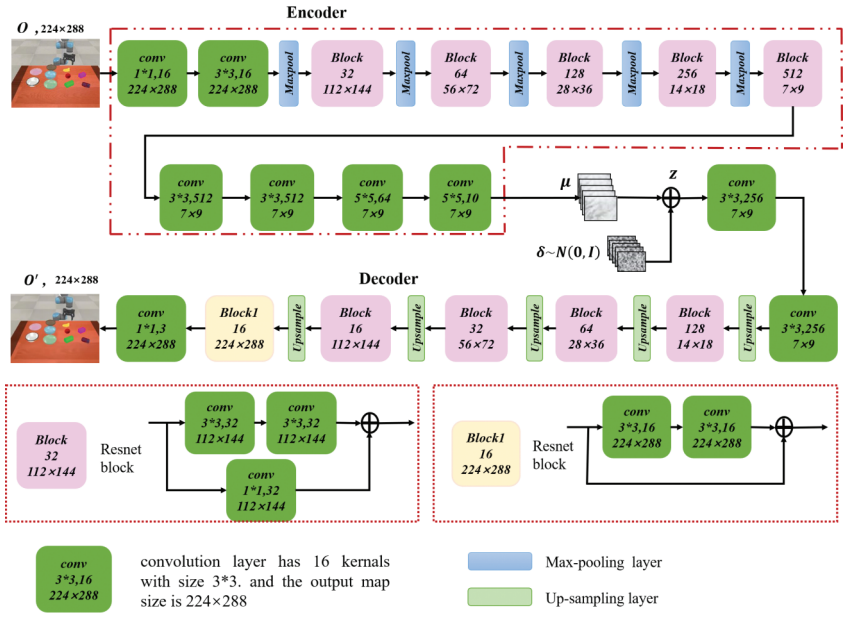


Figure 2 | The detailed structure of variational autoencoder (VAE).

Here, the LSTM is used to encode the series of observations $\boldsymbol{o}$, and the loss function could be changed as follows.

$$\mathcal{L}_{bc} = \|\boldsymbol{\pi}\left(\boldsymbol{E}\left(\boldsymbol{o}\right)_{k-H+1:k}, \boldsymbol{I}\right) - \boldsymbol{a}_k\|_2^2 \tag{12}$$

where $\boldsymbol{H}$ is the length of observation series. As argued before, one-hot vectors ignore the correlations of different tasks, which are very useful for generalization. Composable instructions are explored in this project. Composable instruction means that it consists of several one-hot vectors which could represent different tasks in different combinations. For example, in a simple pick-and-place task, composable instructions could be made up of the object to be picked up and the location to be placed. Instructions represent different tasks or different intents of human beings, which are global attributes. In a normal IL framework, only the next action is predicted by the policy network with current observation or a series of historical observations as described in Equations (11) and (12). Nevertheless, humans not only plan the next action but also plan a series of future actions at the same time, which is called foresight or prospection. Although there are different ways to integrate the prospection with the IL framework, one of the simplest forms is used in the proposed method. The next action as output of LSTM is replaced by a string of future actions. Only the nearest future action is executed, and other future actions are only used to guide the robot to pay more attention to the global features. The loss function could be changed as:

$$\mathcal{L}_{bc} = \sum_{i}^{L} \boldsymbol{\gamma}_i \|\boldsymbol{\pi}\left(\boldsymbol{E}\left(\boldsymbol{o}\right)_{k-H+1:k}, \boldsymbol{I}\right) - \boldsymbol{a}_{k+i}\|_2^2 \tag{13}$$

where $\boldsymbol{\gamma}$ are the weights of different actions and $\boldsymbol{L}$ is the length of prospection. It is found that prospection is very important to learn the purposes of instructions in multi-task scenarios. Prospection makes the robot pay more attention to the global characteristic and obtain the intention of the instruction more easily. Detailed structure of the control network is illustrated in Figure 3. The perception network encodes the observation $\boldsymbol{o}$ into a latent variable $\boldsymbol{\mu}$ of size

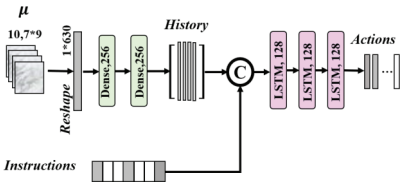


Figure 3 | The structure of control network.

$7 \times 9 \times 10$. After the convolutional encoder, dense layers are followed. However, the dense layers could only receive vector input. The latent variable $\boldsymbol{\mu}$ has to be reshaped into a vector, which's size is 630, before it is inputted into the dense layers. It is then compressed by two dense layers of size 256. The outputs of these layers are concatenated with the vectors of instructions. The inputs to the LSTM are obtained by sampling a series of historical observations based on different length $\boldsymbol{H}$. The outputs of this three-layer LSTMs with a size of 128 are a series of future actions. For robotic manipulation tasks, the control or state information of two-finger gripper is usually considered as a Boolean variable. The loss of cross entropy is used.

$$\mathcal{L}_{gp} = -\left(\boldsymbol{y}\boldsymbol{log}\left(\tilde{\boldsymbol{y}}\right) + \left(\boldsymbol{1} - \boldsymbol{y}\right)\boldsymbol{log}\left(\boldsymbol{1} - \tilde{\boldsymbol{y}}\right)\right) \tag{14}$$

The end of task is also very important for multi-step tasks and instruction-guided manipulation tasks. Therefore, action space is expanded by adding a generalized action STOP that stands for whether the task is finished or not. Nevertheless, the stop state is just in the end of each trajectory in the training data. The ratio of positive and negative samples in the training data is very unbalanced for generalized action. Weighted cross entropy loss is chosen to strengthen the impact of the positive samples.

$$\mathcal{L}_{stop} = -\boldsymbol{\alpha}\boldsymbol{y}\boldsymbol{log}\left(\tilde{\boldsymbol{y}}\right) - \left(\boldsymbol{1} - \boldsymbol{y}\right)\boldsymbol{log}\left(\boldsymbol{1} - \tilde{\boldsymbol{y}}\right) \tag{15}$$

In general, the total loss function of the control network is shown below.

$$\mathcal{L}_{con} = \mathcal{L}_{bc} + \mathcal{L}_{gp} + \mathcal{L}_{stop} \tag{16}$$

4. EXPERIMENT AND RESULT

In order to validate this learning-based end-to-end visuomotor control method, a series of experiments are conducted in a multi-object pick-and-place application scenario in the simulation environment, as illustrated in Figure 4. The simulation setup uses a UR5 robot equipped with an RG2 gripper in V-REP platform [54] with Bullet Physics 2.83 for dynamics, and the remote API functions of V-REP is used for communication with the main Python program. There are five objects to pick up and five containers for placement. The objects are randomly placed in the right half of the table, and the containers placed randomly in the left half of the table, which is illustrated in Figure 5. Therefore, there are totally 25 tasks that represent different object-container pairs. The UR5 robot tries to pick up different objects and then puts them into different containers according to specified instructions. Composable instructions consist of object vectors and container vectors. The simulate RGB camera outputs an image of size 240 × 320, which is rendered with OpenGL. The hardware is a workstation with a 3.7GHz Intel Core i7-8700K CPU and a NVDIA GTX 1080Ti GPU. The operation system is Ubuntu16.04 LTS and the machine learning platform is Tensorflow 1.70.

2500 demonstration trajectories for pick-and-place tasks are collected in V-REP simulation environment, and each task has 100 demonstration trajectories. Images and poses of each trajectory are recorded at a frequency of 10 Hz. Each trajectory lasts six to eight seconds. The demonstrations are divided into two parts and the ratio of each task is 4 to 1. 2000 demonstrations are used as training datasets and the other demonstrations are used as validating datasets. Experiments are conducted to address following questions: (1) Could the robot execute pick-and-place manipulation tasks according to the specified composable instructions effectively based on this end-to-end visuomotor control framework? (2) Does the length $\boldsymbol{H}$ of historical observations improve the performance of the robot system? (3) How does the length $\boldsymbol{L}$ of prospection influence the performance of the robot system? (4) Does combining the visual observation information and the proprioceptive information of robot, such as the pose of the end effector, improve the performance of the method? (5) Does the proposed approach with composable instructions have a good generalization ability?

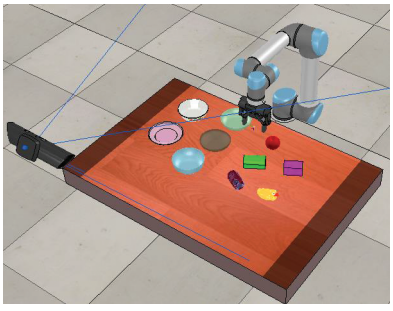


Figure 4 | The multi-object pick-and-place application scenario in V-REP.

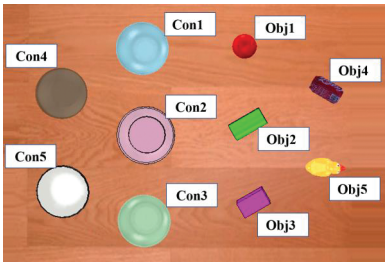


Figure 5 | Objects and containers in pick-and-place manipulation task in V-REP.

4.1. Training Perception Network

As forementioned, the proposed framework mainly includes two modules: a perception network and a control network. They are trained separately in the training phase. Perception network, VAE, is trained firstly and parameters of perception network are fixed while training the control network. The input of VAE is an image of size 224 × 288, and the embedding output is a tensor of size 7 × 9 × 10. Since the original image has a size of 240 × 320, the image is randomly cropped with a window of size 224 × 288. Some data augmentation methods have also been used, including adding salt & pepper noise, flipping and rotating 180 degrees. The actual output of the VAE is the initial cropped image with no noise augmentation. The hyper-parameters $\lambda_1, \lambda_2, \lambda_3$ and λ_4 in Equation (10) are 1, 50, 1000, 10^{-4}, respectively. The VAE is trained using the Adam optimizer ($\beta 1 = 0.9, \beta 2 = 0.999$) with an initial learning rate of 2×10^{-4} and batch-size of 64. The gradients are also cropped in range of −10 to 10. The learning rate is reduced by 10 times every 50 epochs and the network is trained a total of 100 epochs. Perception network is just used to compress high-dimensional perceptual information without making any decisions.

4.2. Training Control Network

The latent tensor is obtained by the trained VAE, where the original image as the input is cropped at the center by a window of size 224×288. Reshaped latent vector is used as part of the control network input and the output is a series of future actions. Parameters of perception network are fixed in the training phase of control network. For the pick-and-place manipulation tasks on the desktop, a four-dimensional action space is selected that includes the position of end effector x, y, z and the angle of rotation θ along z-axis. The weights of the terms in Equation (13) are selected as $0.5, \frac{1}{L}, \ldots \frac{1}{L}$. This means that only the weight of the nearest action is 0.5, and the others are $\frac{1}{L}$. Only the nearest action is executed at each time interval, while other future action sequences are used to make the robot pay more attention to the global characteristic. Hyper-parameter $\boldsymbol{\alpha}$ in $\mathcal{L}_{stop}$ is set to 5 to improve the effect of the positive labels. The control network is trained using the Adam optimizer with an initial learning rate of 1×10^{-3} and batch-size of 10,000. The gradients are cropped in range of −1 to 1. Dropout trick of parameter 0.5 is also used to train the LSTM network. The learning rate is reduced by 2 times every 200 epochs and the network is trained a total of two thousand epochs.

4.3. Testing the Proposed Framework

Two success rates are used to evaluate the performance of control networks with different parameter configurations. One is the success rate of completing the task with specified instruction, and the other is the success rate of picking up target objects. The first one is the main evaluation indicator, which represents the final performance of the proposed method. The successful pickup rate is used as an auxiliary index that could reflect some manipulation capacities of the robot system. For pick-and-place manipulations on the desktop, the pickup action has higher precision requirements while the action of placing objects into bowls has higher execution tolerances. Therefore, tasks are divided into five groups based on the target object to be picked up in the evaluation phase. Traditional one-hot vector-based framework is chosen as the baseline. We replace the composable instruction with a 25-dimensional one-hot vector in the control framework shown in Figure 3. Without prospection, the control network only predicts the next action rather than a series of future actions. Correspondingly, the loss of the BC $\mathcal{L}_{bc}$ is formalized as Equation (12). The length of observation histories $\boldsymbol{H}$ is set to 10 and other hyper-parameters are all the same as that described in Subsection 4.2. For the proposed method, the length of observation histories $\boldsymbol{H}$ is set to 10 and the length of prediction action series $\boldsymbol{L}$ is set to 20. After trained, the robot tries to finish each task group for 25 times based on these two frameworks respectively, and the results are shown in Table 1. "Group 1" represents the task group with target object "Obj1" which needs to be picked up. There are similar meanings for Group 2 to 5. "Avg" means the average success rate of these five groups. "SR1" and "SR2" are the success rate of completing the task with specified instruction and the success rate of picking up target objects. It is validated that the proposed framework has a higher performance and the mechanism of prospection could dramatically improve the effect.

4.4. Different Length of History and Prospection

To explore the effect of length of observation histories $\boldsymbol{H}$ and length of prediction actions $\boldsymbol{L}$, a group of comparative experiments are conducted. Different $(\boldsymbol{H}, \boldsymbol{L})$-pairs are used for training the control network and other hyper-parameters are all shown in Subsection 4.2. The length of observation histories $\boldsymbol{H}$ are set to 2, 5, 10, 20, respectively, and the length of prediction action series $\boldsymbol{L}$ are set to 1, 5, 10, 20, respectively. So, there are a total of 16 $(\boldsymbol{H}, \boldsymbol{L})$-pairs. For each parameter configuration, there are 25 tasks represented by composable instructions, which are also divided into five groups as mentioned above.

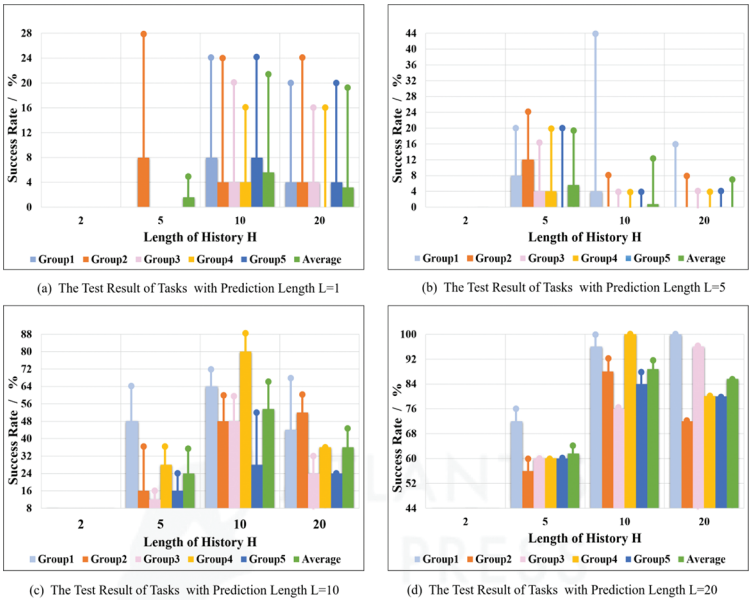
To evaluate the performance of these parameter pairs, 25 trials are performed in each group, and results are shown in Figure 6. Rectangular bar indicates the success rate of task completion. Dot represents the success rate of picking up target objects. It is found that the proposed robotic system has a success rate of zero when the length of observation histories $\boldsymbol{H}$ is two. The robot would get stuck when trying to grasp the target object. Normally, the robot moves very slowly and even pauses for a few time intervals in the grasping phase, and the mapping from observations to actions is multivalued in this case. Training with mean square error (MSE) as the loss function would lead to a false result. Mixed model such as Gaussian mixed model (GMM) may be a solution to eliminate the ambiguity. Increasing the length of observations could be another simple way. The results of different lengths of observations are compared in Figure 6. It is found that results of methods with $\boldsymbol{H}$ of 5 and 10 could significantly improve the performance compared to the result of method with $\boldsymbol{H}$ of 2. However, the method with $\boldsymbol{H}$ of 20 has no improvement compared to the approach with $\boldsymbol{H}$ of 10. And its performance is even slightly worse. The author argues that the observation length $\boldsymbol{H}$ reflects the dimensionality of the input of control network. Proper observation length could eliminate the ambiguity of complex robotic manipulation tasks while larger value of $\boldsymbol{H}$ may increase the risk of overfitting of control network. Besides, prospection also plays an important role in the proposed framework. Comparing Figure 6(a), 6(b), 6(c) and 6(d), it is found that increasing the length of prospection could greatly improve the performance, and the results of method with $\boldsymbol{L}$ of 20 displays the best performance over different lengths of prospection. Under the same prospection length of 20, success rates of different $\boldsymbol{H}$ are 0%, 61.6%, 88.8% and 85.6%, respectively. It is validated that robot could effectively execute manipulation tasks specified by composable instructions based on this end-to-end visuomotor control framework. Both the $\boldsymbol{H}$ and $\boldsymbol{L}$ would affect the performance of the proposed approach, and they are the answers of question 1 to 3.

Table 1 The result of different frameworks.

Group		1	2	3	4	5	Avg
Baseline	SR1	0.08	0.04	0.04	0.08	0.04	0.056
	SR2	0.28	0.20	0.24	0.24	0.20	0.232
Ours	SR1	0.96	0.88	0.76	1.00	0.84	0.888
	SR2	1.00	0.92	0.76	1.00	0.88	0.912

4.5. Combining Observation and Proprioception

Both the observation and the proprioception of the robot are used as inputs to the control network in many visuomotor control methods for robotic manipulation [42,43]. Some experiments are conducted by merging the observation and the proprioception of the robot in this application situation. The structure of changed control network is shown in Figure 7. The position and orientation of the end effector is used as the proprioception information. The compressed latent vectors, the vectors of position and orientation, and the composable instructions are used as inputs of the LSTM module. Length of observation and proprioception are chosen to be 10, 20, respectively. Other hyper-parameters are all the same as shown in Subsection 4.2. Tasks are also divided into five groups, each performing 25 trials, and results are shown in Table 2. "W Pro" means the method with proprioception information of the robot, and "W/o Pro" represents the method without proprioception information which is also shown in Table 1. It is found that inputs by merging observation information and proprioception information of the robot could not improve the performance of proposed end-to-end visuomotor control approach for robotic manipulation tasks. Using the LSTM network to simultaneously learn a time series of images and proprioception information may make the network pay more attention to the series of positions of the end effector, and weaken the influence of a series of observations that contain rich important environmental information.



(a) The Test Result of Tasks with Prediction Length L=1
(b) The Test Result of Tasks with Prediction Length L=5
(c) The Test Result of Tasks with Prediction Length L=10
(d) The Test Result of Tasks with Prediction Length L=20

Figure 6 | Test results with different historical lengths and predicted lengths.

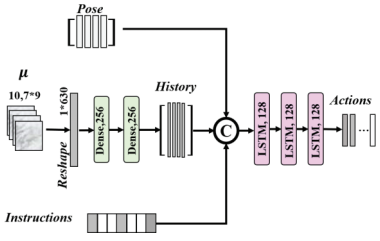


Figure 7 | Control network structure combining observations with proprioceptive information.

Table 2 | The test result of frameworks with/without proprioception information.

Group		1	2	3	4	5	Avg
W/ Pro	SR1	0.48	0.72	0.68	0.56	0.56	0.600
	SR2	0.48	0.72	0.68	0.56	0.60	0.608
W/o Pro	SR1	0.96	0.88	0.76	1.00	0.84	0.888
	SR2	1.00	0.92	0.76	1.00	0.88	0.912

4.6. Generalization of Proposed Method

The generalization ability of the proposed framework is discussed in this subsection. Composable instructions are used for multi-task learning to get interpretable and extensible robot skills in this framework. There are 25 tasks, including 10 basic units, which are 5 target objects and 5 containers. The one-hot vector-based framework is also integrated with prospection mechanism, and it can be obtained by replacing the composable instruction with a 25-dimensional one-hot vector in the control framework shown in Figure 3. For both frameworks, length of observation and proprioception are set to 10, 20, respectively. Demonstrations of 20 tasks are used for training the control network and the left 5 tasks are reserved for testing. The performance of these different networks is shown in Table 3, which includes a network trained with partial datasets and a network trained with full demonstrations. "C/full" represents the framework with composable instruction, which is trained by the demonstrations of the total 25 tasks. "C/part" represents the framework with composable instruction, which is trained by the demonstrations of the 20 tasks. "O/full" and "O/part" mean the framework with one-hot vector-based instruction trained by corresponding demonstrations. "Task 1" represents the task with "Obj1" and "Con1" pairs, and there are similar meanings for Task 2 to 5. "Avg" means the average success rate of these five tasks. We can find that these two frameworks have similar performance in trained tasks. However, for the proposed framework, it is found that the network trained with partial demonstrations could also perform the Task 1 to Task 5 with an average success rate of 91.2%, even if they have not been trained before. The one-hot vector-based framework has no effect on the untrained tasks with a success rate of zero. In conclusion, it can be noticed that the proposed end-to-end visuomotor control framework with composable instruction has good generalization capabilities. Examples of execution of different tasks are shown in Figure 8. For the same surroundings, the robot executed different trajectories depending on different instructions.

Table 3 | The result of different frameworks trained by full/part demonstrations.

Task		1	2	3	4	5	Avg
O/full	SR1	1.00	0.96	0.84	0.92	0.92	0.928
	SR2	1.00	0.96	0.92	0.96	0.92	0.952
O/part	SR1	–	–	–	–	–	–
	SR2	–	–	–	–	–	–
C/full	SR1	0.96	1.00	0.88	0.92	0.96	0.944
	SR2	0.96	1.00	0.96	0.96	0.96	0.968
C/part	SR1	0.96	1.00	0.80	0.80	1.00	0.912
	SR2	1.00	1.00	0.88	0.80	1.00	0.936

5. CONCLUSION AND FUTURE WORK

This paper proposes a method for robotic manipulation by combining composable instructions with learning-based end-to-end visuomotor control framework for multi-task problems. Composable

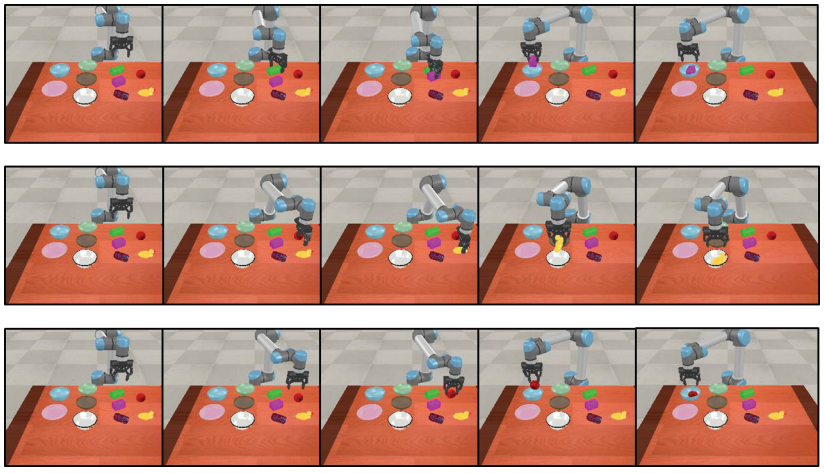


Figure 8 | Trajectories of execution of different tasks based on different instructions.

instructions are used to explore the relations of different tasks via learning the basic units of tasks by itself. Prospection strategy is also used in this framework, which means that the control network predicts not only the next action but also a sequence of future actions simultaneously. Prospection makes the robot pay more attention to the global characteristics and makes it easier to learn the intents of the instructions. Different experimental results demonstrate the effectiveness of this proposed method. Having said that, the application scenario of this method is still relatively simple and it would be explored with more manipulation actions such as pushing, rotating and so on in the future. Long series of high-level instructions is also worth exploring further.

ACKNOWLEDGMENT

This research is mainly supported by Special Program for Innovation Method of Ministry of Science and Technology of China (2018IM020100), National Natural Science Foundation of China (51775332, 51975360, 51975350) and the Cross Fund for medical and Engineering of Shanghai Jiao Tong University (YG2017QN61).

REFERENCES

[1] K. Huebner, S. Ruthotto, D. Kragic, Minimum volume bounding box decomposition for shape approximation in robot grasping, in 2008 IEEE International Conference on Robotics and Automation, IEEE, Pasadena, 2008, pp. 1628–1633.

[2] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, T. Funkhouser, Learning synergies between pushing and grasping with self-supervised deep reinforcement learning, in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Madrid, 2018, pp. 4238–4245.

[3] M. Kennedy, K. Schmeckpeper, D. Thakur, C. Jiang, V. Kumar, K. Daniilidis, Autonomous precision pouring from unknown containers, IEEE Robot. Autom. Lett. 4 (2019), 2317–2324.

[4] W. Meeussen, M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, *et al.*, Autonomous door opening and plugging in with a personal robot, in 2010 IEEE International Conference on Robotics and Automation, IEEE, Anchorage, 2010, pp. 729–736.

[5] J. Hemming, C.W. Bac, B.A.J. van Tuijl, R. Barth, J. Bontsema, E.J. Pekkeriet, E. Van Henten, A robot for harvesting sweet-pepper in greenhouses, in International Conference of Agricultural Engineering, Zurich, 2014. http://edepot.wur.nl/309949

[6] N. Abdo, C. Stachniss, L. Spinello, W. Burgard, Robot, organize my shelves! Tidying up objects by predicting user preferences, in 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2015, pp. 1557–1564.

[7] M.T. Mason, Toward robotic manipulation, Ann. Rev. Control Robot. Auton. Syst. 1 (2018), 1–28.

[8] S. James, A.J. Davison, E. Johns, Transferring end-to-end Visuomotor control from simulation to real world for a multi-stage task, in Conference on Robot Learning, Mountain View, 2017, pp. 334–343. http://proceedings.mlr.press/v78/james17a.html

[9] T. Lampe, M. Riedmiller, Acquiring visual servoing reaching and grasping skills using neural reinforcement learning, in The 2013 International Joint Conference on Neural Networks (IJCNN), IEEE, Dallas, 2013, pp. 1–8.

[10] S. Lange, M. Riedmiller, A. Voigtländer, Autonomous reinforcement learning on raw visual input data in a real world application, in The 2012 International Joint Conference on Neural Networks (IJCNN), IEEE, Brisbane, 2012, pp. 1–8.

[11] I. Lenz, H. Lee, A. Saxena, Deep learning for detecting robotic grasps, Int. J. Robot. Res. 34 (2015), 705–724.

[12] S. Levine, *et al.*, Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, Int. J. Robot. Res. 37 (2018), 421–436.

[13] S. Gu, E. Holly, T. Lillicrap, S. Levine, Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates, in 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Singapore, 2017, pp. 3389–3396.

[14] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, S. Levine, Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration, in 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Brisbane, 2018, pp. 3758–3765.

[15] P. Abolghasemi, A. Mazaheri, M. Shah, L. Bölöni, Pay attention!-Robustifying a deep visuomotor policy through task-focused attention, arXiv preprint arXiv:1809.10093, 2018.

[16] D. Xu, S. Nair, Y. Zhu, J. Gao, A. Garg, L. Fei-Fei, S. Savarese, Neural task programming: learning to generalize across hierarchical

tasks, in 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Brisbane, 2018, pp. 1–8.
[17] C. Ye, Y. Yang, *et al.*, What can I do around here? Deep functional scene understanding for cognitive robots, in 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Singapore, 2017, pp. 4604–4611.
[18] M. Schwarz, A. Milan, A.S. Periyasamy, S. Behnke, RGB-D object detection and semantic segmentation for autonomous manipulation in clutter, Int. J. Robot. Res. 37 (2018), 437–451.
[19] A. Eitel, J.T. Springenberg, L. Spinello, M. Riedmiller, W. Burgard, Multimodal deep learning for robust RGB-D object recognition, in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Hamburg, 2015, pp. 681–687.
[20] M. Schwarz, H. Schulz, S. Behnke, RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features, in 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Seattle, 2015, pp. 1329–1335.
[21] J. Miseikis, K. Glette, O.J. Elle, J. Torresen, Automatic calibration of a robot manipulator and multi 3d camera system, in 2016 IEEE/SICE International Symposium on System Integration (SII), IEEE, Sapporo, 2016, pp. 735–741.
[22] D. Song, C.H. Ek, K. Huebner, D. Kragic, Task-based robot grasp planning using probabilistic inference, IEEE Trans. Robot. 31 (2015), 546–561.
[23] R. Alterovitz, S. Koenig, M. Likhachev, Robot planning in the real world: research challenges and opportunities, Ai Mag. 37 (2016), 76–84.
[24] Y. Zhang, S. Sreedharan, A. Kulkarni, T. Chakraborti, H.H. Zhuo, S. Kambhampati, Plan explicability for robot task planning, in Proceedings of the RSS Workshop on Planning for Human-Robot Interaction: Shared Autonomy and Collaborative Robotics, Ann Arbor, 2016.
[25] M. Nieuwenhuisen, D. Droeschel, D. Holz, J. Stückler, A. Berner, J. Li, R. Klein, S. Behnke, Mobile bin picking with an anthropomorphic service robot, in 2013 IEEE International Conference on Robotics and Automation, IEEE, Karlsruhe, 2013, pp. 2327–2334.
[26] A. Jain, C.C. Kemp, Pulling open novel doors and drawers with equilibrium point control, in 2009 9th IEEE-RAS International Conference on Humanoid Robots, IEEE, Paris, 2009, pp. 498–505.
[27] K. Yamazaki, Y. Watanabe, K. Nagahama, K. Okada, M. Inaba, Recognition and manipulation integration for a daily assistive robot working on kitchen environments, in 2010 IEEE International Conference on Robotics and Biomimetics, IEEE, Tianjin, 2010, pp. 196–201.
[28] M. Hersch, F. Guenter, S. Calinon, A. Billard, Dynamical system modulation for robot learning via kinesthetic demonstrations, IEEE Trans. Robot. 24 (2008), 1463–1467.
[29] Y. Demiris, A. Dearden, From motor babbling to hierarchical learning by imitation: a robot developmental pathway, in Proceedings of the Fifth International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems Lund University Cognitive Studies, Nara, 2005, pp. 31–37. http://cogprints.org/4961/
[30] J. Kober, J.A. Bagnell, J. Peters, Reinforcement learning in robotics: a survey, Int. J. Robot. Res. 32 (2013), 1238–1274.
[31] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, G. Cottrell, Understanding convolution for semantic segmentation, in 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, Lake Tahoe, 2018, pp. 1451–1460.
[32] J. Han, D. Zhang, G. Cheng, N. Liu, D. Xu, Advanced deep-learning techniques for salient and category-specific object detection: a survey, IEEE Signal Process. Mag. 35 (2018), 84–100.
[33] L. Pinto, A. Gupta, Supersizing self-supervision: learning to grasp from 50k tries and 700 robot hours, in 2016 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Stockholm, 2016.
[34] Z. Zeng, Z. Zhou, Z. Sui, O.C. Jenkins, Semantic robot programming for goal-directed manipulation in cluttered scenes, in 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Brisbane, 2018, pp. 7462–7469.
[35] S. Levine, N. Wagener, P. Abbeel, Learning contact-rich manipulation skills with guided policy search, in 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Seattle, 2015, pp. 156–163.
[36] S. Levine, C. Finn, T. Darrell, P. Abbeel, End-to-end training of deep visuomotor policies, J. Mach. Learn. Res. 17 (2016), 1334–1373. http://jmlr.org/papers/v17/15-522.html
[37] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, P. Abbeel, Asymmetric actor critic for image-based robot learning, arXiv preprint arXiv: 1710.06542, 2017.
[38] C. Finn, X.Y. Tan, Y. Duan, T. Darrell, S. Levine, P. Abbeel, Deep spatial autoencoders for visuomotor learning, in 2016 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Stockholm, 2016, pp. 512–519.
[39] M.A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, J. Bohg, Making sense of vision and touch: self-supervised learning of multimodal representations for contact-rich tasks, in 2019 International Conference on Robotics and Automation (ICRA), Montreal, 2019.
[40] T. Zhang, Z. McCarthy, O. Jowl, D. Lee, X. Chen, K. Goldberg, P. Abbeel, Deep imitation learning for complex manipulation tasks from virtual reality teleoperation, in 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Brisbane, 2018, pp. 1–8.
[41] F. Torabi, G. Warnell, P. Stone, Behavioral cloning from observation, in Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence Main Track, Stockholm, 2018.
[42] C. Finn, T. Yu, T. Zhang, P. Abbeel, S. Levine, One-shot visual imitation learning via meta-learning, arXiv preprint arXiv: 1709.04905, 2017.
[43] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, S. Levine, One-shot imitation from observing humans via domain-adaptive meta-learning, arXiv preprint arXiv: 1802.01557, 2018.
[44] S. James, M. Bloesch, A.J. Davison, Task-embedded control networks for few-shot imitation learning, in 2nd Conference on Robot Learning (CoRL), Zurich, 2018.
[45] K. Suzuki, H. Mori, T. Ogata, Motion switching with sensory and instruction signals by designing dynamical systems using deep neural network, IEEE Robot. Auto. Lett. 3 (2018), 3481–3488.
[46] K. Hausman, J.T. Springenberg, Z. Wang, N. Heess, M. Riedmiller, Learning an embedding space for transferable robot skills, in International Conference on Learning Representations, Vancouver, 2018.
[47] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, in ICLR, Banff, 2014.

[48] H. Van Hoof, N. Chen, M. Karl, P. van der Smagt, J. Peters, Stable reinforcement learning with autoencoders for tactile and visual data, in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Daejeon, 2016, pp. 3928–3934.
[49] A. Hämäläinen, K. Arndt, A. Ghadirzadeh, V. Kyrki, Affordance learning for end-to-end Visuomotor robot control, arXiv preprint arXiv: 1903.04053, 2019.
[50] A.J. Piergiovanni, A. Wu, M.S. Ryoo, Learning real-world robot policies by dreaming, arXiv preprint arXiv: 1805.07813, 2018.
[51] L. Ma, J. Chen, Y. Liu, Using RGB image as visual input for mapless robot navigation, arXiv preprint arXiv: 1903.09927, 2019.
[52] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, 2016, pp. 770–778.
[53] M.-Y. Liu, T. Breuel, J. Kautz, Unsupervised image-to-image translation networks, in Advances in Neural Information Processing Systems, Long Beach, 2017, pp. 700–708.
[54] E. Rohmer, S.P.N. Singh, M. Freese, V-REP: a versatile and scalable robot simulation framework, in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, Tokyo, 2013, pp. 1321–1326.