One-Shot Domain-Adaptive Imitation Learning via Progressive Learning Applied to Robotic Pouring

Dandan Zhang, *Member, IEEE*, Wen Fan, *Graduate Student Member, IEEE*, John Lloyd, Chenguang Yang, *Senior Member, IEEE*, and Nathan F. Lepora, *Member, IEEE*

***Abstract*— Traditional deep learning-based visual imitation learning techniques require a large amount of demonstration data for model training, and the pre-trained models are difficult to adapt to new scenarios. To address these limitations, we propose a unified framework using a novel progressive learning approach comprised of three phases: i) a coarse learning phase for concept representation, ii) a fine learning phase for action generation, and iii) an imaginary learning phase for domain adaptation. Overall, this approach leads to a *one-shot domain-adaptive imitation learning* framework. We use robotic pouring as an example task to evaluate its effectiveness. Our results show that the method has several advantages over contemporary end-to-end imitation learning approaches, including an improved success rate for task execution and more efficient training for deep imitation learning. In addition, the generalizability to new domains is improved, as demonstrated here with novel backgrounds, target containers, and granule combinations in the experiment. We believe that the proposed method is broadly applicable to various industrial or domestic applications that involve deep imitation learning for robotic manipulation, and where the target scenarios are diverse and human demonstration data is limited. For project video, please check our website: `https://sites.google.com/view/imitation-learning-tase2022`.**

***Note to Practitioners*—The motivation of this paper is to develop a progressive learning framework, which can be used for both service and industrial robots to learn from human demonstrations, and then transfer the learned skill to different scenarios with ease. We use the robotic pouring task as an example to demonstrate the effectiveness of our proposed method, since pouring is an essential skill for service robots to assist humans' daily lives, and can benefit robot automation in wet-lab industries. The aim of this research is to enable robots to obtain visuomotor skills (such as the pouring skill), and accomplish the tasks with a high success rate using our proposed progressive learning method. We conducted experiments to show that the proposed method has good performance, high data efficiency and evident generalizability. This is significant for intelligent robots working in various practical applications.**

Manuscript received 17 July 2022; accepted 20 October 2022. This article was recommended for publication by Associate Editor X. Zhong and Editor J. Li upon evaluation of the reviewers' comments. *(Corresponding author: Dandan Zhang.)*

Dandan Zhang, Wen Fan, John Lloyd, and Nathan F. Lepora are with the Department of Engineering Mathematics, University of Bristol, BS8 1TH Bristol, U.K., and also with the Bristol Robotics Laboratory, BS34 8QZ Bristol, U.K. (e-mail: ye21623@bristol.ac.uk).

Chenguang Yang is with the Department of Engineering Design and Mathematics, University of the West of England, BS16 1QY Bristol, U.K., and also with the Bristol Robotics Laboratory, BS34 8QZ Bristol, U.K.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TASE.2022.3220728.

Digital Object Identifier 10.1109/TASE.2022.3220728

***Index Terms*— Efficient robot skill learning, imitation learning, robot and automation, one-shot learning, robotic pouring.**

I. Introduction

IMITATION learning is an effective tool for robots to learn dexterous manipulation skills [1], [2], [3], [4], [5], in scenarios where obtaining a dynamic model for control or specifying a reward function [6] for reinforcement learning are challenging. However, a large database is normally required for training control policies [7]. Moreover, the policies trained in a specific environment may not work well in other environments, due to the presence of domain gaps. An ideal automatic robotic manipulation system should be able to adapt to new scenarios for task execution even if only very limited demonstration data is available for training the control policy [8]. To this end, we develop a **one-shot domain adaptive imitation learning** framework that is data-efficient and can generalize the learned behavior to a new scenario with novel domain characteristics without significant loss in performance.

Humans are good at learning and generalizing strategies for everyday tasks from a few demonstrations. In particular, humans can learn the concepts for performing a series of tasks and then transfer that knowledge to new scenarios by practicing specific visuomotor skills. Motivated by the advantages of human learners, we propose a **progressive learning** method that decouples the traditional end-to-end imitation learning pipeline into three phases: coarse learning, fine learning and imaginary learning. Our method allows the robot to acquire the general knowledge with a good concept representation in the coarse learning phase [9], then learn to generate the precise motions in the fine learning phase, and finally expand this knowledge to new scenarios in the imaginary learning phase, which mimics the progressive learning process that humans also appear to do.

We use robotic pouring as an example task to evaluate the effectiveness of the proposed method, which is an essential skill for industrial or domestic robots when used for dispensing lubricants [10], carrying out chemical experiments [11], [12], cleaning [13], and cooking [14]. We chose this task because pouring involves complex dynamic processes that are difficult to model [15]. Moreover, it is not feasible for robots to learn from trial-and-error based on reinforcement learning approaches for the pouring task, because of the large amount

of human intervention that would be required while training the robot. To this end, we consider the robotic pouring task as an appropriate example to validate the proposed imitation learning framework.

The **main contributions** of this paper are as follows.

1) We train the robot to learn general concepts by encoding concept representation features during the **coarse learning phase**, which provides compact but interpretable features extracted from raw pixels. This paves the way for the robot to learn action generation with high efficiency.

2) We enable the robot to generate precise motions using an LSTM-Attention hybrid model during the **fine learning phase**, based on the features extracted at the coarse learning phase, which ensures the success rate of task execution by incorporating concept representation with temporal information.

3) We employ a generative adversarial network to generate a large amount of synthetic observation data in new scenarios during the **imaginary learning phase**, which enhances the perception skills of the robot and ensures that the robot can adapt the pre-trained policies to new scenarios with ease.

In summary, we formulate a one-shot domain-adaptive imitation learning framework and demonstrate a progressive learning approach that can implement such framework. The proposed method addresses the fundamental limitations of deep imitation learning by eliminating the need for recollecting a large amount of demonstration data and retraining the whole model in new domains with unseen object properties or environments.

The rest of this paper is structured as follows. Section II introduces the related work. The problem statement and the construction of the proposed framework are introduced in Section III. Following that, we use a robotic pouring as a concrete example to demonstrate the proposed progressive learning approach in Section IV. The experiment design and result analysis are described in Section V. Finally, conclusions are drawn in Section VI.

II. Related Work

A. Imitation Learning

Imitation learning considers the problem of acquiring skills from observing demonstrations. Behaviour Cloning and Inverse Reinforcement Learning (IRL) are two major research directions for imitation learning [16], [17]. Behaviour Cloning aims to teach the robot to follow the expert guidance from supervised learning. IRL estimates a reward function from human demonstration, and then the learned reward function is used for reinforcement learning. Survey articles include [18], [19], [20].

For traditional imitation learning, a large amount of demonstration data is normally required for training the policies [7]. Moreover, policies obtained during the model training phase are domain-specific [21]. For example, a model trained for task execution in one scene through an imitation learning algorithm in the training phase may not result in good performance for task execution in the testing phase when encountering a new scenario with new properties. Therefore, recent research for imitation learning has been focusing on one-shot learning and domain adaptation, which enhances the data efficiency and generalizability of traditional imitation learning approaches.

B. One-Shot Imitation Learning

One-shot imitation learning enables robots to learn to perform a new task with few demonstrations from humans [22], [23], [24], [25]. The objective of one-shot imitation learning is to train action prediction networks that are not specific to one task, then maximize the expected performance of the learned policy when faced with a new, previously-unseen task. In one example, a one-shot imitation learning method uses one demonstration and the observed state as the input and generates the action value for a block stacking task [26]. The first vision-based one-shot imitation learning framework [27] used Model-Agnostic Meta-Learning (MAML) [28] for an object placing task. However, this method requires the training database to have high diversity with many demonstration trials for different tasks; for example, about 1300 demonstrations were collected for meta-training during the training phase [27]. Therefore, in this paper we aim to eliminate the need for collecting a large amount of diverse demonstration data during the training phase. Moreover, to verify that the one-shot learning can be implemented for more complex tasks, we evaluate our proposed method on a robotic pouring task, which involves more complex dynamic processes.

C. Domain Adaptation

Domain adaptation is a process that allows a neural network model trained with samples from a source domain to generalize to a target domain [29]. Recent domain adaptation methods learn deep neural transformations that map image data from dstinct domains into a common feature space. For example, adversarial discriminative domain adaption (ADDA) [30] is a method for domain adaptation in image classification. However, the labels in the source domain and the target domain were required to be identical, which is unrealistic for imitation learning in robotics and does not align with the one-shot imitation learning setting. Another research direction is to reconstruct the target domain from the source representation [31]. To this end, a generative adversarial network can be used to generate a large database with synthetic observation data for model training to support domain adaptation [32], which we will use in this paper.

D. Learning-Based Robotic Pouring

Previously, imitation learning approach has been investigated for a dynamic fluid pouring task similar to the one in this paper [33]. However, that previous study required many failed human demonstrations for the robot to learn how to recover from errors [33]. In another work, an RNN-enabled MPC (Model Predictive Control) controller determined the optimal velocity for pouring task execution, which was verified with a custom apparatus. A single motor was used to generate the rotational motions [34], instead of a robotic arm. While the approach was able to learn the task, the generalizability was not demonstrated, in particular the 3D position of the source

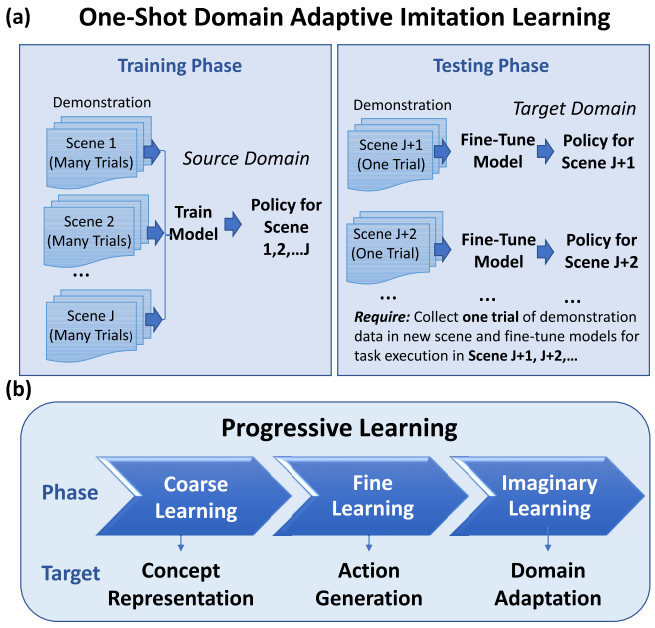

Fig. 1. Schematic diagram of the proposed learning framework and approach. (a) Illustration of the one-shot domain-adaptive imitation learning framework. (b) The workflow of the progressive learning method, including a coarse learning phase, a fine learning phase, and an imaginary learning phase.

container cannot be adjusted. Thus, the above methods can perform well in specific scenarios, but there are open questions about how well they would carry over to new scenarios with novel target containers, granules and backgrounds. To this end, learning-based method for robotic pouring task with generalizability will be developed and evaluated in this paper.

III. Methodology: Progressive Learning

A. Overview

Our goal is to learn a policy that can adapt to new domains from a single demonstration of that task in a new scenario during the testing phase, while eliminating the need for collecting a large amount of data from different scenes during the training phase (in contrast to traditional one-shot learning, as discussed in Section II-B).

As shown in Fig. 1(a), the data collection process is similar to that of traditional imitation learning, with several trials of demonstration data collected for several scenes (Scene 1, 2,$\cdots$, J). During the testing phase, for task execution in Scene $J+1$, $J+2$, $\cdots$, only one trial of demonstration data is required. The model obtained in the training phase can be fine-tuned during the testing phase and new policies for task execution in new scenes can be obtained quickly.

Our approach is to develop a progressive learning approach, which may be considered as a representative architecture for the one-shot domain-adaptive imitation learning scheme. The workflow of our progressive learning approach is shown in Fig. 1(b).

B. Problem Formulation

We use $\boldsymbol{o_t}(t = 1, 2, \ldots, T)$ to represent the observation and $\boldsymbol{a_t}(t = 1, 2, \ldots, T)$ to represent the action at time t. Then τ represents a trajectory for performing the task, consisting a sequence of observation and action pairs: $\tau = [(\boldsymbol{o}_1, \boldsymbol{a}_1), (\boldsymbol{o}_2, \boldsymbol{a}_2), \ldots, (\boldsymbol{o}_T, \boldsymbol{a}_T)]$.

1) Training Phase: Let $\mathcal{T}_j = \{\tau_1, \tau_2, \cdots, \tau_K\}$ denote, let sb do sth a group of trajectories for task execution in Scene j. Let $\mathcal{D}_\mathcal{T} = \{T_1, T_2, \cdots, T_J\}$ denote the database for model training, which is comprised of different trajectories $\tau_k(k = 1, 2, \cdots, K)$ for task execution in different scenes $T_j(j = 1, 2, \cdots, J)$, demonstrated by human for the robot to imitate. A policy $\boldsymbol{\pi}_\Theta$ will be obtained to map observations $\boldsymbol{o_t}$ to actions $\boldsymbol{a_t}$ under the parameters Θ. We denote the observation data used for model training in the training phase as being from source domain $\boldsymbol{H}$.

2) Testing Phase: Let $\mathcal{D}^r_\mathcal{T} = \{T_{J+1}, T_{J+2}, \cdots\}$ denote the new database collected during the testing phase. $T_{J+1}, T_{J+2}, \cdots$ represent new scenarios for task execution, which have novel domain characteristics unseen during the training phase. Unlike the training phase (where each scene T_j has several demonstrated trajectories), $\mathcal{D}^r_\mathcal{T}$ only includes a single trajectory collected as demonstration data for each scene $T_{J+1}, T_{J+2}, \cdots$, while the corresponding policies $\pi_{\Theta_{J+1}}$, $\pi_{\Theta_{J+2}}, \cdots$ for task execution are obtained via transfer learning. We denote the observation data collected during the testing phase as being from target domain $\boldsymbol{R}$.

C. Framework Construction

1) Coarse Learning - Concept Representation: The main goal of the coarse learning phase is to enable the robot to learn basic concepts by encoding representation from raw pixels, which can be used to accelerate the action generation model training during the fine learning phase while ensuring model performance by preserving compact but interpretable features.

Representation learning techniques aim to extract features from high-dimensional sensory input, which can help improve the model performance in some downstream learning tasks [35]. Though deep neural networks such as VGG16, InceptionV3, and ResNet50 [36] with pre-trained weights on ImageNet can be used for transfer learning, the encoded features cannot explicitly express the contexts for a specific task. Auto-encoders and their variances can be used for feature extraction, but these models cannot guarantee the representation is human-interpretable. Therefore, we aim to train a model that can learn concept representation with interpretable features that benefit the downstream process (action generation) when data is limited.

Let $\boldsymbol{F_C(.)}$ denote the coarse learning model. In this paper, the architecture of the coarse learning model $\boldsymbol{F_C(.)}$ is an adapted version of a ResNet-18 model [37]. Different from the original ResNet-18 model for single image classification, we reorganize the architecture into a multi-head structure for multi-variable classification. During model training, we draw a batch of samples $\{X_k, Y_k\}(k = 1, 2, \ldots, K)$ to update the parameters of $\boldsymbol{F_C(.)}$ in a supervised learning manner, with batch size K, inputs $X_k = \boldsymbol{o_t}$ and outputs Y_k. The construction of the key components of Y_k will be detailed in Section IV-E ('coarse learning phase').

After $\boldsymbol{F_C(.)}$ is trained using database $\mathcal{D}_\mathcal{T}$, we use it to convert the image data to their corresponding concept

representation feature vectors. A new database $\mathcal{D}'$ is then constructed in which the original observation-action pairs $\tau_k = [(\boldsymbol{o}_1, \boldsymbol{a}_1), (\boldsymbol{o}_2, \boldsymbol{a}_2), \ldots, (\boldsymbol{o}_T, \boldsymbol{a}_T)]$ are replaced by state-action pairs $\tau'_k = [(\boldsymbol{s}_1, \boldsymbol{a}_1), (\boldsymbol{s}_2, \boldsymbol{a}_2), \ldots, (\boldsymbol{s}_T, \boldsymbol{a}_T)]$ with $\boldsymbol{s_t} = V_F^t$. This database will be used for the fine learning process next.

2) Fine Learning - Action Generation: The fine learning process aims to utilize the general concepts obtained by the coarse learning phase to generate the precise action during task execution.

We use a Long-Short Term Memory (LSTM) [38], which contains memory cells and gates that allow the network to propagate gradients back in time. The network takes the inputs at the current time step alongside hidden states from previous time steps to generate the output for the current time step. LSTM models are good at processing sequential data, and will thus be used as an essential architecture to construct a fine learning model of the action values.

A potential issue with the LSTM model is that it may overfit by memorizing the mean trajectory, which makes it harder to generalize to novel tasks. To address this issue with the vanilla LSTM architecture, we incorporate an attention mechanism [39]: instead of considering all neighbors as equal, the neighboring neurons are weighted according to a criterion specified by the attention model.

The architecture of the fine learning model, denoted $\boldsymbol{F_f(.)}$, consists of two LSTM layers with 128 and 64 units respectively, following the standard described in [40]. A key aspect of our model is to improve upon previous work using LSTM-based motion control for robotic pouring, by incorporating an attention mechanism [41]. After combining the LSTM model and the attention layer, the generated encoded features pass through a multi-layer perceptron with four hidden units to predict the actions for task execution.

After obtaining the extracted concept representation feature vector of each frame, a sequence of feature vectors can be formed by $\boldsymbol{V_F^w} = [V_F^{t-w+1}, V_F^{t-w+2}, \ldots, V_F^{t-1}, V_F^t]$, where the parameter w sets the length of the sequence. The fine learning model uses the features obtained from observation images at these times $t-w+1$, $t-w+2$, $\ldots$, $t-1$, t as input, with the robot's end-effector action values as output. During model training, we draw a batch of samples $\{X_k, Y_k\}(k = 1, 2, \ldots, K)$ to update the parameters of $\boldsymbol{F_f(.)}$ in a supervised learning manner, where K represents the batch size.

3) Imaginary Learning - Domain Adaptation: The Pix2Pix GAN [42] has been used for domain adaptation from the simulated environment to a real environment [43]. However, aligned image pairs from different domains are required for the model training, which is not realistic for robotic dexterous manipulation task because the duration of end-effectors' trajectories in different scenarios may vary significantly. Alternatives such as the CycleGAN [44], DiscoGan [45] or DualGan [46] can achieve image translation between different domains using unpaired images in an unsupervised learning manner. In this paper, we use CycleGAN to transfer images from the original database to a new database that includes sufficient synthetic data with the appropriate properties (domain characteristics) for generalization to novel scenarios. We note in passing that it should be possible to achieve similar levels of performance using other types of GAN model such as DiscoGAN and DualGAN [45], [46].

Our proposed method will benefit robotic manipulation tasks that are either too difficult to model or too costly to learn from failures via reinforcement learning approaches. Unlike prior methods for domain adaptation that require time-aligned and paired demonstrations from different domains to obtain state correspondences, our proposed method enables the robot to adapt to new scenarios via imaginary learning in an unsupervised manner.

The imaginary learning phase can be implemented by three steps: i) training the generators via CycleGAN, ii) constructing the synthetic database, and iii) transferring knowledge via fine-tuning. In Step 1, the CycleGAN model is trained in an unsupervised learning manner. However, in Step 3, the model is fine-tuned in a supervised learning manner.

Step 1: **Training Generators via CycleGAN.** First, we collect one-shot demonstration data for task execution in a novel scene with new domain characteristics. This database is denoted as $\mathcal{D}^r_{\mathcal{T}}$.

We sample observation data from $\mathcal{D}_{\mathcal{T}}$, which is drawn from the source domain $\boldsymbol{H}$, and then sample observation data from $\mathcal{D}^r_{\mathcal{T}}$, which is drawn from the target domain $\boldsymbol{R}$. For domain adaptation, we need to train a generator $\boldsymbol{G(.)}$ to take observation images from the source domain and generate new observation images matching those from the target domain, while a generator $\boldsymbol{G'(.)}$ takes the images back from the target domain to the source domain. Meanwhile, a discriminator $\boldsymbol{D_H(.)}$ is trained to classify whether a data sample is drawn from the source or from the generated data in the target domain, while a discriminator $\boldsymbol{D_R(.)}$ is trained to classify whether a data sample is drawn from the target or generated data in the source domain. The distribution of the real observations remains fixed, and the distribution of the generating observation is learned to match the real data. The aim is to solve a min-max problem such that

$$G^*, G'^* = \arg\min_{G,G'} \max_{D_H,D_R} \mathcal{L}\big(G, G', D_H, D_R\big). \tag{1}$$

The details of the loss functions for training the CycleGAN can be found in Appendix.

Step 2: **Constructing the Synthetic Database.** The next step is to use the generator $\boldsymbol{G(.)}$ to generate new observation data in an imaginary manner to construct a synthetic database $\mathcal{D}^{r'}_{\mathcal{T}}$. More specifically, we sample trajectories $\tau_1, \tau_2, \cdots = [(\boldsymbol{o_1}, \boldsymbol{a_1}), (\boldsymbol{o_2}, \boldsymbol{a_2}), \cdots]$, and generate new observation images to replace the original observation images from the original trajectories as new trajectories $\tau'_1, \tau'_2, \cdots = [(\boldsymbol{G(o_1)}, \boldsymbol{a_1}), (\boldsymbol{G(o_2)}, \boldsymbol{a_2}), \cdots]$. These trajectories are considered to be imaginary demonstration data with generated observation and action pairs, giving $\mathcal{D}^{r'}_{\mathcal{T}} = \{\tau'_j\}(j = 1, 2, \cdots)$ comprised of a series of imaginary trajectories.

Step 3: **Transferring Knowledge via Fine-Tuning.** Finally, a new database $\mathcal{D}^o_{\mathcal{T}}$ is constructed by combining $\mathcal{D}^{r'}_{\mathcal{T}}$ and $\mathcal{D}^r_{\mathcal{T}}$, which is used to fine-tune $\boldsymbol{F_C(.)}$ in a transfer learning manner. Subsequently, we combine f_p (extracted by $\boldsymbol{F_C(.)}$ with updated parameters) and the new task-specific characteristics z' to generate a new feature vector V_F^o. In this manner, $\mathcal{D}^{o'}_{\mathcal{T}}$

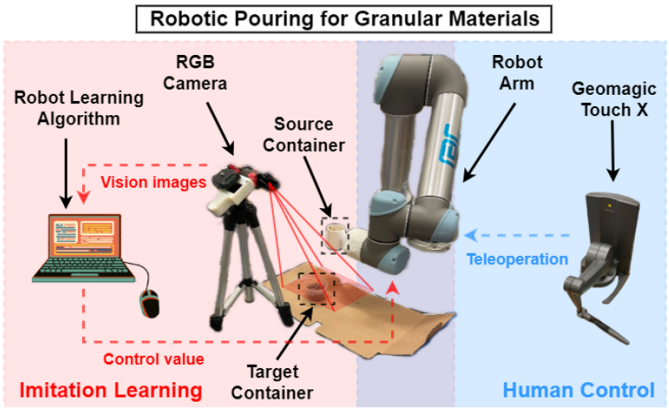

Fig. 2. The experimental setup for the robotic pouring task. A pouring container is mounted as the end effector to the wrist of a UR5 (Universal Robots) 6-axis robot arm. An RGB camera is mounted at a fixed position to view a target container placed on a table adjacent to the base of the arm. Human demonstration data is collected via teleoperation using Geomagic Touch X haptic motion-capture device as the remote controller.

is constructed by pairing V_F^o with the corresponding action values. After fine-tuning $\boldsymbol{F_f(.)}$ using $\mathcal{D}_{\mathcal{T}}^{o'}$, a series of generated new policies $\pi_{\Theta_{J+1}}$, $\pi_{\Theta_{J+2}}$, $\cdots$ are obtained for task execution in novel scenarios with new domain characteristics absent from the original training database $\mathcal{D}_{\mathcal{T}}$, achieving the appropriate domain adaptation.

IV. Case Study: Robotic Pouring

A. Task Description

The pouring task requires the robot to successfully pour different granular materials into different target containers with distinct background environments. Here we consider distinct scenes to correspond to cases where either granular materials, target containers, or backgrounds are different from previously experienced combinations. The robot should learn to adjust a reasonable position of the source container and control the wrist motions in a proper manner to avoid spilling the materials out of the target containers. More specifically, the robot needs to learn 4-DoF end-effector control and ensure a reasonable success rate for pouring granular materials to target containers with different shapes in different environments. We collected human demonstration data to train a model for online deployment. The learned model will generate action values to command the robot to execute the pouring task. The robot then returns to its original starting configuration after each episode of the pouring task is completed.

B. Hardware Deployment

The human demonstration database for model training was collected via teleoperation of a UR5 (Universal Robots) 6-axis robot arm, using a Geomteric Touch X (3D Systems) haptic motion-capture device as the remote controller to provide human-guided commands. An RGB camera sited to view the container was used to capture the image frames for training. The source container was attached as an end effect to the wrist of the robotic arm, with the target container placed on a table near the base of the robot arm. The experimental setup for the robotic pouring task is shown in Fig. 2.

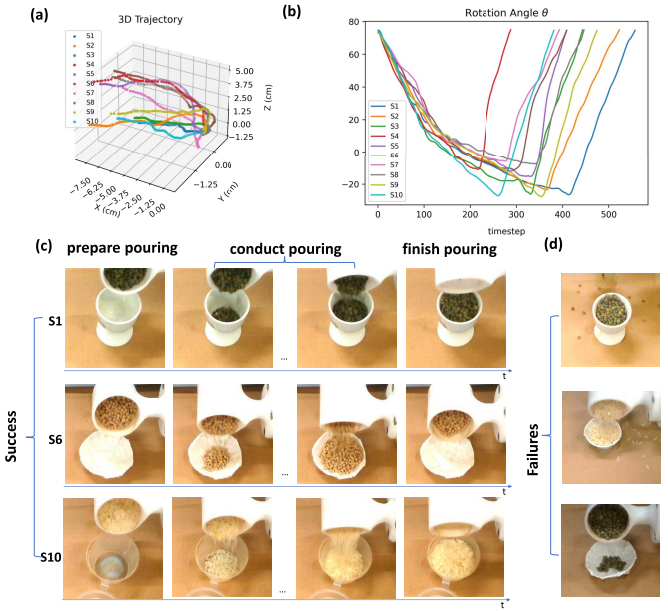

Fig. 3. Overview of the constructed database for model training. (a) Raw trajectories of the 3D position of the end-effector. ('S' is the abbreviation of 'Scene') (b) Raw trajectories of the rotation angle of the robot's wrist joint. (c) Examples of three successful trails of the pouring experiments. (d) Examples of failed trails.

C. Database Construction

The demonstration database is constructed from ten distinct pouring scenes (known as scene S1-S10), as shown in Fig. 4. Eight trials are collected for each scene respectively. The properties of the target containers, granules and background used for different pouring scene are summarized in Table I. Four different types of target containers are used for experiments: 'goblet', 'plate', 'cup' and 'jar' that have various properties, sizes, and colors. As for the granules, we use, to align with context 'lentils', 'rice', and 'couscous', which have different colors and shapes. A sheet of orange cardboard is used as the background during data collection, which will be changed during testing.

The database contains a series of trajectories that are comprised of observation-action pairs. This training database, denoted as $\mathcal{D}_{\mathcal{T}}$, will be used for model training in the coarse learning phase and fine learning phase. The images captured as observation data $\boldsymbol{o_t}$ during teleoperation are cropped to size of 224×224 at 30 fps. The robot kinematics states are recorded simultaneously as the corresponding action values. To pour the granules, the tilt angle of the source container is controlled directly by commanding the rotating angle $\theta(t)$ of the wrist joint of the robotic arm, while the 3-dimensional (3D) position $p(t) = [x(t), y(t), z(t)]$ of the end-effector is adjusted to ensure pouring without spillage. We use the 3D velocity $v(t) = [v_x(t), v_y(t), v_z(t)]$ and rotation angle for robot control during online deployment. Therefore, we define action values as $\boldsymbol{a_t} = [v_x(t), v_y(t), v_z(t), \theta(t)]$ in this paper.

D. Performance Evaluation

The success rate is used to evaluate the performance of the imitation learning algorithm, which is a ratio between the

TABLE I
DETAILS OF THE DATABASE

Scene	Target Container				Granules			Background		Trails		Evaluation
	Type	Property	Size	Color	Type	Color	Shape	Color	Type	Train	Test	
1	Goblet	Opaque	Medium	White	Lentils	Green	Oblate	Orange	Board	8	1	
2	Goblet	Opaque	Medium	White	Rice	White	Prolate	Orange	Board	8	1	
3	Goblet	Opaque	Medium	White	Couscous	Yellow	Cylindrical	Orange	Board	8	1	✓
4	Plate	Opaque	Small	White	Lentils	Green	Oblate	Orange	Board	8	1	✓
5	Plate	Opaque	Small	White	Rice	White	Prolate	Orange	Board	8	1	
6	Plate	Opaque	Small	White	Couscous	Yellow	Cylindrical	Orange	Board	8	1	
7	Cup	Opaque	Small	White	Lentils	Green	Oblate	Orange	Board	8	1	
8	Cup	Opaque	Small	White	Rice	White	Prolate	Orange	Board	8	1	✓
9	Jar	Transparent	Big	/	Lentils	Green	Oblate	Orange	Board	8	1	
10	Jar	Transparent	Big	/	Rice	White	Prolate	Orange	Board	8	1	✓
11	Cup	Opaque	Small	White	Couscous	Yellow	Cylindrical	Orange	Board	0	1	
12	Jar	Transparent	Big	/	Couscous	Yellow	Cylindrical	Orange	Board	0	1	
13	Jar	Transparent	Big	/	Lentils	Green	Oblate	Blue	Tissue	0	1	✓(New Background)
14	Plate	Opaque	Small	White	Lentils	Green	Oblate	Blue	Tissue	0	1	(New Background)
15	Goblet	Opaque	Medium	White	Lentils	Green	Oblate	Blue	Tissue	0	1	(New Background)
16	Cup	Opaque	Small	White	Coffee	Brown	Irregular	Orange	Board	0	1	✓(New Granules)
17	Plate	Opaque	Small	White	Coffee	Brown	Irregular	Orange	Board	0	1	(New Granules)
18	Goblet	Opaque	Medium	White	Coffee	Brown	Irregular	Orange	Board	0	1	(New Granules)
19	Cup	Opaque	Big	Black	Rice	White	Prolate	Orange	Board	0	1	✓(New Container)
20	Cup	Opaque	Big	Black	Couscous	Yellow	Cylindrical	Orange	Board	0	1	(New Container)

✓ represents that the scene will be used for physical experiments for model evaluation.

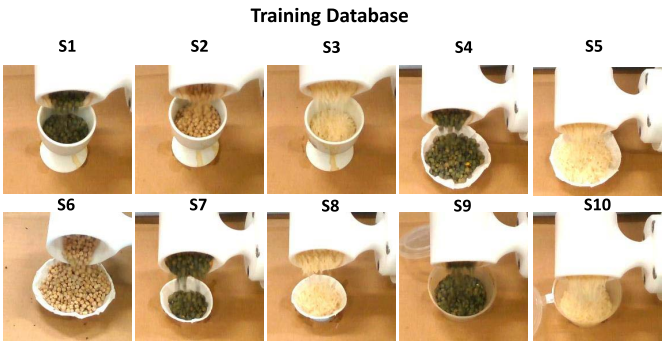

Fig. 4. Examples of the ten different pouring scenes, S1-S10, for constructing the training database ('S' abbreviates 'Scene').

number of successful trials and the total conducted during the experiments. A successful trial is defined as one in which the granules are poured from the source container into the target container without spilling. If the total volume of the granules in the source container is smaller than the maximum capacity of the target container, a successful trial is defined as pouring at least 90% total volume of granules from the source container to the target container. If the maximum capacity of the target container is smaller than the source container, the target container should be filled to at least 90% of the total capacity of the target container.

Ten raw trajectories, including the 3D position and the rotation angle of the wrist joint of the robot's end-effector, are plotted in Fig. 3(a)-(b) as examples. Fig. 3(c) shows three successful trials of robotic pouring as examples, while Fig. 3(d) indicates failures when executing the pouring task.

E. Implementation Details

The neural network architecture of the progressive learning method is shown in Fig. 5. The implementation details of automatic robotic pouring using the proposed method are illustrated as follows.

1) Coarse Learning - Concept Representation: The goal of the coarse learning phase is to obtain a concept representation feature vector that includes information from several key components in an interpretable manner.

The concept representation of the pouring task is comprised of three key components: i) the tilt angle control, ii) the 3D-position adjustment, and iii) the task-specific characteristics (encoded as a distinct variable z).

***Component 1:* Tilt Angle Control.** Imagine that a 'teacher' (human demonstrator) is responsible for supervising a 'student' (robot imitator) to learn the pouring task by transferring general concepts using descriptive language. The whole pouring process could then be split into several stages, such as:

(1) increase the tilt angle of the source container quickly until granules flow out from the source container;

(2) keep increasing the tilt angle of the source container stably to fill up the target container;

(3) reduce the tilt angle of the source container when the target container is almost full or no remaining granules are in the source container.

Suppose that θ_s is the tilt angle of the source container when the granules begin to leave the container. Then θ_m represents the maximum tilt angle, after which the robot starts to restore to its original pose. The tilt angle of the source container is the rotation angle of the robot's wrist (axis-6), denoted by $\theta(t)$.

For a simple implementation, the first stage of the task could be labelled as $\widetilde{\theta}(t) = 0$ (when $\theta(t) < \theta_s$), the second and the third stages labelled as $\widetilde{\theta}(t) = 1$ (when $\theta_s \leq \theta(t) \leq \theta_m$) and $\widetilde{\theta}(t) = 2$ (when $\theta_m < \theta(t)$) respectively. The supervision for tilt angle control could then be formulated as a 3-class classification problem on the visual images.

For a more general implementation, we want the 'student' to learn detailed knowledge, such as 'increase the tilt angle slightly' or 'increase the tilt angle significantly'. In this case, the second stage could be further segmented into many sub-stages. Suppose that there are N stages in total with the first stage is denoted as $\widetilde{\theta}(t) = 0$ and the final stage denoted as

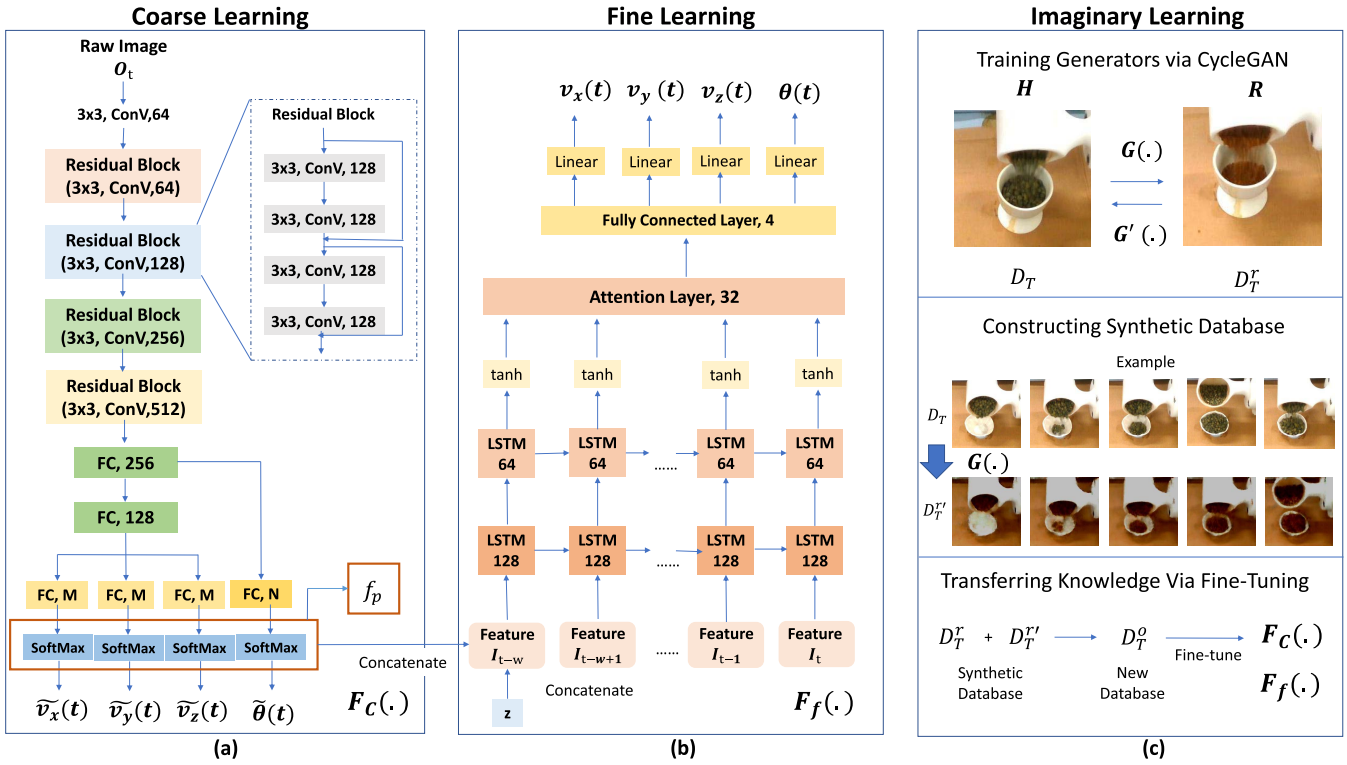

Fig. 5. The architecture of the progressive learning method. (a) The architecture of the coarse learning model for feature extraction, (b) The architecture of the fine learning model for action generation, (c) The workflow illustration of the imaginary phase for domain adaptation.

$\widetilde{\theta}(t) = N - 1$. Then, we define the discrete class labels as

$$\widetilde{\theta}(t) = \begin{cases} 0 & \text{if } \theta(t) < \theta_s \\ \lceil (\theta(t) - \theta_s)/\theta_r \rceil & \text{if } \theta_s \le \theta(t) \le \theta_m \\ N - 1 & \text{if } \theta_m < \theta(t), \end{cases} \quad (2)$$

where $\theta_r = (\theta_s - \theta_m)/(N-2)$ represents a normalization to the original simpler implementation during the second stage. In consequence, the class label $\widetilde{\theta}(t)$ for a specific observation o_t is integer-valued, and the supervision for tilt angle control is formulated as an N-class classification problem.

Component 2: **3D Position Adjustment.** During each stage, to avoid the granules missing the target container, the 'teacher' should instruct the 'student' to adjust the 3D position of the source container in a reasonable manner. For example, if the source container is far away from the target container, the 'teacher' can give instructions to the student like 'move forwards', 'move backwards', 'move left', 'move right', 'move up', 'move down' to control the source container to reach a desired position for pouring that depends on the relative 3D-position between the two containers. If the source container has already been located at a reasonable position, the instruction can be simply to 'keep still'.

Suppose that $v_q(t)(q = x, y, z)$ represent the end-effector's linear velocity along the x-, y- and z-axis respectively. Then we denote $v_s > 0$ as a threshold value on intentional motion, such that if $||v_q(t)|| < v_s$, then the velocity is regarded as not caused by a human's intention to move but by unintentional motion, e.g. tremor, during the teleoperation. For the simplest implementation, $\widetilde{v}_q(t) = 0$ when $v_s \le v_q(t)$ and $\widetilde{v}_q(t) = 2$ when $v_q(t) \le -v_s$, representing motions in the positive and negative directions along the x-, y-, or z-axis respectively; meanwhile, $\widetilde{v}_q(t) = 1$ when $||v_q(t)|| < v_s$ represents not moving. In this case, the supervision for 3D-position control can be formulated as a 3-class classification problem.

For a more general implementation, we want the 'student' to learn both the direction and magnitude for 3D-position adjustment, such as to 'move quickly in the positive direction' or 'move slowly in the positive direction'. Suppose that there are M-classes of instructions for position adjustment in total, then the class labels are defined as:

$$\widetilde{v}_q(t) = \begin{cases} 0 & \text{if } v_s \le v_q(t) \\ M - 1 & \text{if } v_q(t) \le -v_s \\ \lceil (v_q(t) + v_s)/m \rceil & \text{if } -v_s < v_q(t) < v_s \end{cases} \quad (3)$$

where $m = 2v_s/(M-2)$ is a normalization to give integer class labels.

Component 3: **Task-Specific Characteristics Encoding.** In addition, a 'teacher' may also give task-specific information to the 'student' that could include physical factors that affect the pouring dynamics. For example, if the granules/liquids used for pouring have high friction, then the pouring velocity should be higher to ensure they flow quickly out of the source container. Conversely, if the target container is small in size, then the velocity for adjusting the 3D-position of the source container should be relatively slow to minimize spilling the granules/liquids out of the target container.

Here a variable z is used to represent the characteristics of the task, relating to factors that affect the pouring dynamics but do not change with time. In this work, $z = [C, P]$, where C is a scalar that indicates the size of the target container (depending

on its maximum capacity) and P is a scalar that indicates the type of granules used for pouring. Here we used three distinct types of granules in the experiments, denoted by 3 scalar values ($P = 0,1,2$ for lentils, rice and couscous). In the coarse learning phase, $X_k = \boldsymbol{o_t}$, $Y_k = [\widetilde{v_x}(t), \widetilde{v_y}(t), \widetilde{v_z}(t), \widetilde{\theta}(t)]$.

Concept Representation: Since the first and second component can be formulated as a classification problem, we use categorical cross-entropy loss $L_p(p = 0, 1, 2, 3)$ to update the parameters of the neural network model by maximizing the accuracy of prediction of the distinct variables $\widetilde{\theta}(t)$, $\widetilde{v_x}(t)$, $\widetilde{v_y}(t)$, $\widetilde{v_z}(t)$. The overall loss function L is the linear combination of L_p with different weights. For the pouring task, tilt angle control is more important and has higher relationship with the success rate of task execution. We set the weight as 0.4,0.2,0.2,0.2 for L_p respectively in this paper.

When the model training is completed, a "softmax" output layer is then used to learn each of the distinct outputs. After obtaining the probability distribution of each output variable, all four probability distributions are concatenated into a single feature vector $f_p(t)$ of dimension $3M + N$. Subsequently, this feature vector $f_p(t)$ is concatenated with the contextual representation z to give an overall feature vector $V_F^t = [f_p(t), z]$, which is obtained for each image frame $\boldsymbol{o_t}$.

Here we will use the class probabilities produced by the coarse learning model as 'soft inputs' to train the fine learning model (see below), instead of using the 'hard inputs' (one-hot encoded values) [47]. Our reasoning is that the 'soft inputs' have high entropy and thus contain more information than 'hard inputs', because they not only provide information on the most probable class but also on the other classes according to their probabilities.

The neural network model for coarse learning is trained via self-supervision, since the labels can be generated automatically based on (2),(3), which eliminates the need of manual annotation of data. We don't need to label data for concept representation in the coarse learning phase.

2) Fine Learning - Action Generation: In the fine learning phase, $X_k = \boldsymbol{V_F^w}$, $Y_k = [\theta(t), v_x(t), v_y(t), v_z(t)]$. To ensure that the generated velocities are safe and reasonable for controlling the robot, we calculate the mean and variance of the angular velocity $\omega(t) = \dot{\theta}(t)$ for every pouring stage determined by (2). We also obtain the lower and upper bounds, ω_{min} and ω_{max}, of the angular velocity of the wrist joint's rotation during pouring. We then use these statistics to form a safety constraint to ensure that the generated angular velocity remains within $[\omega_{min}, \omega_{max}]$ during online deployment.

The loss function $\mathcal{L}_r$ $(r = \{v_x, v_y, v_z, v_\theta\})$ for each component of the predicted action uses a Mean Square Error (MSE) between the predicted and target outputs. The overall loss function is a sum of the MSE over the four distinct outputs.

After training the models $F_C(.)$ and $F_f(.)$ using the demonstration database in the source domain $\boldsymbol{H}$, the policy $\boldsymbol{\pi_\Theta}$ can be used for the pouring task in scenarios involving target containers, granules, and background that have been seen before. The domain adaptation in the testing phase is implemented via imaginary learning, which is described next.

3) Imaginary Learning - Domain Adaptation: First, we collect one-shot demonstration data for task execution in a novel scene with new domain characteristics (e.g., a new background, a different container or a different granular material). According to Table I, scenes S13-S15 have a new background (blue tissue), scene S16-S18 include new granules (brown coffee with irregular shapes), scene S19-S20 include a new container (large black cup), none of which has been demonstrated previously in the training phase. A key aspect of our method is that only one trial of demonstration data need be collected during the testing phase for each of these scenes S13-20, with this database denoted $\mathcal{D}_\mathcal{T}^r$. We then sample observation data from $\mathcal{D}_\mathcal{T}$ as source domain $\boldsymbol{H}$ and sample observation data from $\mathcal{D}_\mathcal{T}^r$ as target domain $\boldsymbol{R}$. Following that, we train the $\boldsymbol{G}(.)$ based on (1) for new data generation. Fig. 6 shows several examples of image generation using CycleGAN-based techniques to transfer the original image data to synthetic data in the new domain.

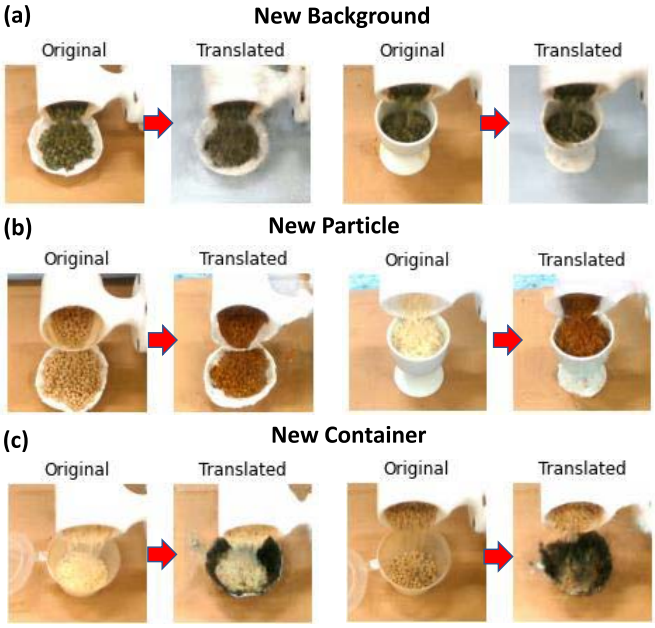

Fig. 6. Examples of the results for domain adaptation based on a CycleGan. Image generation to (a) new backgrounds (blue tissue), (b) new granules (coffee with irregular shapes and lentils with red color), and (c) a new target container (big black cup and while cup).

An overall summary of the workflow for progressive learning is given in **Algorithm 1**.

V. Experiments and Results

A. Experiment Design

After initial model training described above, we conducted real-time experiments on a UR5 robot arm, using the setup shown in Fig. 2. Three distinct groups of experiments were conducted in total to answer three research questions, which can be known as ablation study for coarse learning, fine learning and imaginary learning respectively.

- Whether the concept representation features extracted during the coarse learning phase enhance the training efficiency of the action generation model during the fine learning phase or not?

Algorithm 1 Progressive Learning

Input: Demonstration Database $\mathcal{D}_{\mathcal{T}}$
Required: learning rate α; batch size K;
Coarse Learning Phase;
Initialize parameters Θ for $\boldsymbol{F_C(.)}$;
while *Training* $\boldsymbol{F_C}(\Theta, \mathcal{D}_{\mathcal{T}})$ **do**
  Sample batch of trajectories from $\mathcal{D}_{\mathcal{T}}$;
  Convert $\theta(t)$ to $\widetilde{\theta}(t)$ based on (2);
  Convert $v_q(t)$ to $\widetilde{v}_q(t)$ based on (3);
  Construct $\{X_k, Y_k\}(k = 1, 2, \ldots, K)$ for training;
  Compute loss function L;
  Backpropagate gradient of $\boldsymbol{F_C(.)}$;
  Update parameters Θ;
end
Construct new database $\mathcal{D}'$ using $\boldsymbol{F_C(.)}$;
Fine Learning Phase;
Initialize parameters Θ_r of $\boldsymbol{F_f(.)}$;
while *Training* $\boldsymbol{F_f}(\Theta_r, \mathcal{D}')$ **do**
  Sample batch of trajectories τ_k from $\mathcal{D}'$;
  Construct sequential data based on τ_k:
  $\boldsymbol{V_F^w} = [V_F^{t-w+1}, V_F^{t-w+2}, \ldots, V_F^{t-1}, V_F^t]$;
  Construct $\{X_k, Y_k\}(k = 1, 2, \ldots, K)$ for training;
  Compute loss function $\mathcal{L}_r$;
  Backpropagate gradient of $\boldsymbol{F_f(.)}$;
  Update parameter Θ_r;
end
Imaginary Learning Phase;
Construct database $\mathcal{D}^r_{\mathcal{T}}$ with one-shot demonstration;
Draw observation data from $\mathcal{D}_{\mathcal{T}}$ as source domain $\boldsymbol{H}$;
Draw observation data from $\mathcal{D}^r_{\mathcal{T}}$ as target domain $\boldsymbol{R}$;
Obtain the optimal $\boldsymbol{G(.)}$ based on (1);
(see **Appendix**)
Sample $\tau_{j(j=1,2,\cdots)}$=$[(\boldsymbol{o_1}, \boldsymbol{a_1}), (\boldsymbol{o_2}, \boldsymbol{a_2}), \cdots]$;
Generate new trajectories:
$\tau'_j(j = 1, 2, \cdots) = [(\boldsymbol{G}(\boldsymbol{o_1}), \boldsymbol{a_1}), (\boldsymbol{G}(\boldsymbol{o_2}), \boldsymbol{a_2}), \cdots]$;
Construct $\mathcal{D}^{r'}_{\mathcal{T}}$ with $\tau'_j(j = 1, 2, \cdots)$;
$\mathcal{D}^o_{\mathcal{T}} \leftarrow \mathcal{D}^r_{\mathcal{T}} \cup \mathcal{D}^{r'}_{\mathcal{T}}$;
Fine-tune $\boldsymbol{F_C(.)}$ using $\mathcal{D}^o_{\mathcal{T}}$;
Construct new database $\mathcal{D}^{o'}_{\mathcal{T}}$;
Fine-tune the parameters of $\boldsymbol{F_f(.)}$ using $\mathcal{D}^{o'}_{\mathcal{T}}$;
Output: Model $\boldsymbol{F_C(.)}$,$\boldsymbol{F_f(.)}$.

- How does the performance of the action generation model obtained during the fine learning phase compare with the traditional behavior cloning method in terms of the success rate of the pouring task?
- Novel domain characteristics represent the usage of new backgrounds, new types of granules, new target containers that have not been included in the database for model training. Can the algorithm demonstrate its generalizability in the imaginary learning phase by performing the pouring task in new scenarios with novel domain characteristics?

With the experiments mentioned above, we can prove the significance of each component for the proposed progressive learning.

TABLE II
RESULTS FOR COMPARISONS BETWEEN WITH AND WITHOUT COARSE LEARNING AND THE ANALYSIS OF DATA EFFICIENCY

Model & Data	Training	Testing
Without + 100%Data	0.0036	0.0037
With + 100%Data	0.0020	0.0023
With + 50%Data	0.0022	0.0023
With + 25%Data	0.0024	0.0029

B. Coarse Learning Model Enhances the Data Efficiency

The first group of experiments evaluates the effectiveness of the concept representation features for just the coarse learning phase of the overall framework.

An offline analysis is conducted first to evaluate the performance of the coarse learning model. For an ablation study, we use the one-hot values as features to replace the original concept representation features, so that this comparison study is conducted between with and without concept representation features.

When using the whole database without concept representation features for training the action generation model in the coarse learning phase, the training and testing MSE are 0.0036 and 0.0037 respectively. These are markedly poorer than the training and testing MSE for our proposed method using the extracted features, which are 0.0020 and 0.0023 respectively. If 50% of data from the original database is used for model training, then the training and testing MSE become 0.0022 and 0.0023 respectively, and with only 25% of data from the original database for model training, the MSE become 0.0024 and 0.0029 respectively. The results are summarized in Table II.

These results show that when using the coarse learning model for concept representation features extraction, the performance of the action generation is better than that without, even if we only use 25% of the available data. Thus, we conclude that the coarse learning model enhances the data efficiency of the action generation model training.

C. Fine Learning Model Improves the Success Rate

The second group of experiments compares the proposed approach with a baseline method which implements imitation learning using one model. The comparison is conducted in terms of the success rate for automatic robotic pouring in four distinct scenes.

We used an end-to-end behaviour cloning approach [48] for comparison with our proposed progressive learning method to demonstrate the value of fine learning. The neural network architecture for the baseline model is shown in Fig. 7(c).

Scenes S3, S4, S8 and S10 are selected to cover all the types of granular materials and containers in the experiments. Our evaluation is based on the success rate of the pouring for the different scenes.

Fig. 8 shows four example successful trials of pouring in these scenes, while Table III summarizes the experimental results on the comparison study. The overall results indicated that with our progressive learning method, the success

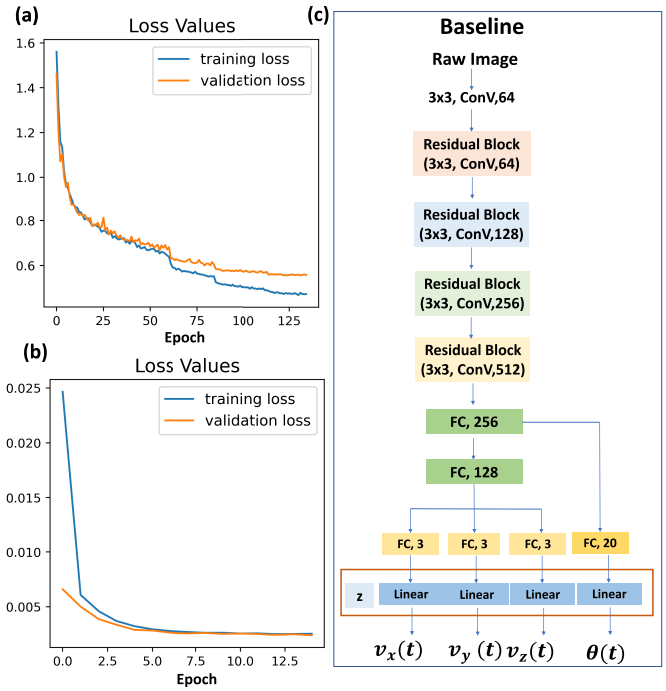

Fig. 7. The training and validation loss for (a) the coarse learning model, (b) the fine learning model, (c) baseline model for comparison between with and without fine learning.

TABLE III

EXPERIMENTAL RESULTS FOR COMPARISONS BETWEEN WITH AND WITHOUT FINE LEARNING

	S3	S4	S8	S10	Mean
Without	2/10	4/10	2/10	6/10	35.0%
With	6/10	8/10	7/10	10/10	77.5%

rate is improved significantly (77.5% vs. 35.0%). Table III demonstrates that our proposed method outperforms the traditional end-to-end learning-based model for action generation.

In our view, the reason why the progressive learning method performs better than the baseline method on this task is that the baseline does not utilize temporal information, resulting in the predicted trajectories being unstable and causing the granules to flow out of the container during the pouring process. That said, the success rate for the distinct scenes has different variances, with the success rate across scenes using the progressing learning method ranging from 60% to 100%. This seems to be due to the fact that some containers have a relatively large opening diameter, and therefore less requirement for precise motion generation during pouring. Since some of the target containers have relatively small opening diameters, the pouring task becomes more challenging, and thus leads to a higher risk of failure. For those target containers with smaller opening diameters, there is potential for improving the task execution success rate by using more advanced computer vision techniques and enhancing the control precision degree.

Another inherent advantage of progressive learning is its ability to adapt to new scenes quickly. The training and validation loss for the coarse learning model $\boldsymbol{F_C(.)}$ and the fine learning model $\boldsymbol{F_f(.)}$ are visualized in Fig. 7 (a) and (b) respectively. The training of $\boldsymbol{F_C(.)}$ requires more than 100 epochs, while the training of $\boldsymbol{F_f(.)}$ only requires 10-20 epochs. The model training in the fine learning process is much faster than the one in the coarse learning process. When applied the proposed method to new environment with variances, we can fine-tune the model obtained in fine learning process without retraining the whole model like the end-to-end learning approach.

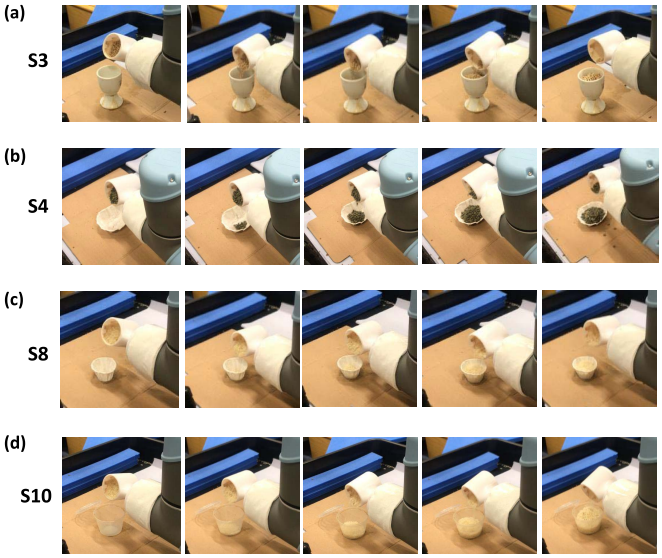

Fig. 8. Examples of four successful pouring trials. The experiments were conducted in (a) scene S3; (b) scene S4; (c) scene S8; and (d) scene S10 according to Table I.

TABLE IV

EXPERIMENTAL RESULTS FOR COMPARISONS BETWEEN WITH AND WITHOUT DOMAIN ADAPTATION

Type of Experiment	Without	With
New Background	2/8	7/8
New Granules	2/8	6/8
New Target Container	3/8	6/8
Mean	29.1%	79.2%

D. Imaginary Learning Ensures the Generalizability

This third group of experiments is aimed at verifying the generalizability of the progressive learning method. These experiments examine whether the proposed progressive learning method can be adapted to new scenarios with new domain characteristics or not, including new environments (backgrounds), granular materials, and target containers that were not included in the demonstration data collection process for model training.

Examples of four successful trials are shown in Fig. 9, including testing the performance of the method when adapted to new backgrounds, new granules, new target containers, as well as the combination of new granules and target container. The experimental results are summarized in Table IV. With the domain adaptation method, the average success rate for the task execution in new scenarios is increased by a large margin from 29.1% to 79.2%. Comparing the success rate with and without domain adaptation, the results indicate that with the adaptation coupled with progressive learning, the robot is able to perform automatic pouring in novel scenarios and has demonstrated good generalizability.

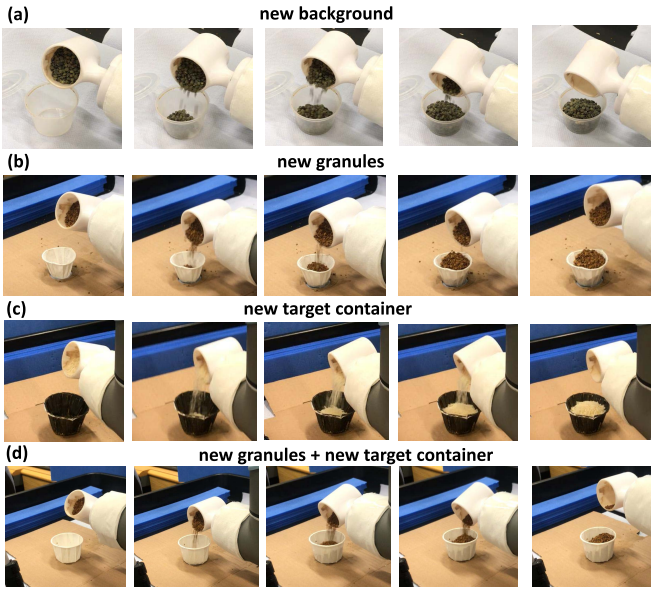

Fig. 9. Four successful trials for demonstrating the generalizability of the proposed method. Domain adaptation to (a) new background, (b) new granules, (c) new target container, and (d) new granules and new target container.

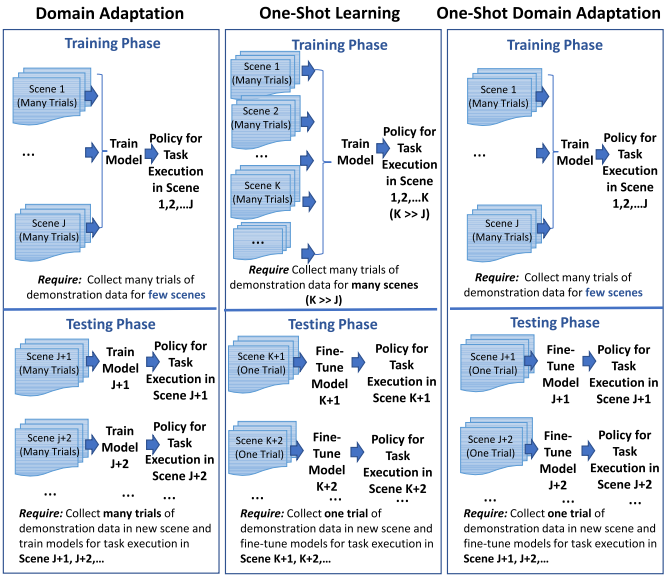

Fig. 10. Concept illustration of different robot learning schemes, including 'domain adaptation', 'one-shot learning' and 'one-shot domain adaptation'.

E. Comparisons With Other Work

The work closest to ours is in [49], where a CycleGAN [44] is used to translate human demonstrations to robot-looking ones at pixel space. An important difference to this work is that we do not aim to learn a reward function for reinforcement learning and do not require the robot to practice the skill to learn its physical execution. Moreover, in [49], the robot needs to query the human user to indicate success or failure at some critical points during the learning process. We try to avoid intensive human supervision in this work, and avoid using reinforcement learning that requires trial-and-error for the robot to learn the action generation policy.

To the best of our knowledge, this is the first time that a one-shot domain adaptation has been defined for imitation learning tasks. The one-shot domain-adaptive imitation learning is different from traditional one-shot learning implemented in [26], [27], [28]. For example, traditional one-shot learning requires the training database to be of high diversity with many demonstration trials for different tasks. For example, about 1300 demonstrations were collected for meta-training during the training phase for [27]. In our paper, we only need to collect 80 demonstrations, which reduced the need for expensive data collection in the training process. In the testing phase, we collect only one trial of demonstration data for domain adaptation, which is similar to traditional one-shot learning. Therefore, the main advantage of 'One-Shot Domain Adaptation' is that it removes the need to collect a much larger, diverse training set during the training phase. We illustrate the difference between the different robot learning schemes in Fig. 10.

We compare our proposed progressive learning method with the other methods in terms of performance, data efficiency and generalizability. We assume the algorithm is of high efficiency if the performance does not decrease significantly after reducing 75% of data for training the model in new scene. If the robot can accomplish a new pouring task with either new environments (backgrounds), granular materials, or target containers that haven't been included in the training dataset, then the applied method can be known to have good generalizability.

TABLE V
COMPARISONS WITH RELATED WORK: "✓" AND "X" INDICATE WITH AND WITHOUT THE RELEVANT PROPERTIES, RESPECTIVELY

Methods	Performance		Data Efficiency	Generalizability
	MSE	SR		
BC-LSTM [34]	0.0023	0.10	✓	X
BC-DCNN [7]	0.0158	/	X	X
BC-ResNet [15]	0.0037	0.35	X	X
Proposed	0.0023	0.78	✓	✓

"/" indicates not applicable; SR indicates success rate.

Table V shows the comparisons of the proposed method with other related work that can be applied to the automatic pouring task, including deep imitation learning method (also known as behavior cloning) based on LSTM, deep convolutional neural network (DCNN) [7], deep residual neural network (ResNet) respectively [15]. We can conclude that our proposed method has the advantage of high success rate, data efficiency and generalizability.

F. Discussion

1) Advantages of the Proposed Method: Compared to other methods, the proposed method can work better when given limited training data. The potential reason is that the robot mimics the progressive learning process that humans appear to do. For example, the robots learn the key concepts in the coarse learning phase, then learn to generate the precise motions in the fine learning phase, and finally learn by analogy (imaginary) by expanding the acquired knowledge to new scenarios. This hierarchical learning process enables the robot to learn with higher efficiency.

The robot can learn an effective feature extraction model during the course learning phase, while the model training in the fine learning process is much faster than the one in the coarse learning process. When applying the proposed method to a new environment with variances, we can fine-tune the models obtained in the fine learning process without retraining the complex model with a large amount of new data. More importantly, CycleGAN is used to generate a large amount of synthetic observation data in new scenarios during the imaginary learning phase, which enhances the perception skills of the robot and ensures that the robot can adapt the pre-trained policies to new scenarios with ease.

2) Adaptation to Other Tasks: The purpose of this work is to provide a general framework that can bring benefits to automation in both service and industrial robotics. While our experiments focus on a robotic pouring task, our framework is not specific to this task, and could also be used for a wide variety of similarly dexterous tasks, spanning from making consumer beverages to handling dangerous chemical solvents, building a pyramid of cups, washing dishes, opening doors. Take the door opening task as an example, in the coarse learning phase, we can replace the component of 'tilt angle control' with the 'door opening angle control', while the component of 3D position adjustment remains the same. As for the task characteristics, we can replace the various types of granules with the various types of door handles' shapes. The fine learning and imaginary learning phase can remain the same.

Moreover, the proposed framework could be modified to more complex scenarios that require the robot to learn from demonstration based on sensory input with multiple modalities. For example, we use image data as observations in this work, while force/torque sensors, tactile sensors, RGB-D/stereo cameras could be incorporated into the proposed progressive learning approach to provide more comprehensive and accurate information as observations, further enhancing the efficiency and generalizability of robot learning.

3) Future Work: To probe limitations of the approach, we notice that for the adaptation to new backgrounds, the lighting condition may influence slightly the performance of the model. In particular, for a background material with high reflectivity, the success rate of the pouring task was reduced, which we attribute to the strong reflected light leading to poor performance of the robot's perception system. Moreover, were the granules to have the same color as the target container, then the pouring task may become more challenging. In the future, we will enhance the robustness of the proposed method by incorporating more sensory information with effective feature extraction techniques.

The proposed progressive learning framework includes three essential modules, each of which can be further improved. For example, we can use more efficient representation learning approach to generate the concept representation for downstream tasks, which can further accelerate the training speed of the fine learning model and enhance its performance. As for fine learning, we can employ Neural Architecture Search (NAS) to optimize the neural network model, which has high potential to outperform manually designed models.

Imaginary learning is the key step that enables domain adaptation to new scenarios. CycleGAN is deployed to generate new data in this paper. The integration of CycleGAN in our proposed progressive learning method is novel, since it enables imaginary learning for adaptation to unseen scenarios. However, a current limitation is that we need to retrain the generator when a new scenario shows up. To overcome this problem, other types of GANs such as Lifelong GAN [50] can be used in future development to update the generator continuously when adapted to a new task.

VI. Conclusion

In this paper, we proposed a progressive learning framework to achieve one-shot domain-adaptive imitation learning. The robotic manipulator learned a concept representation in the coarse learning phase, and then progressed to learning how to generate accurate actions for task execution across multiple pouring scenarios in the fine learning phase. To improve the generalization to new domains, imaginary learning was implemented via CycleGAN to generate new observations for the robot to enhance its perception capability in new scenarios during the imaginary learning phase.

Our experiments were based on a robotic pouring task that required the robot to pour different types of granular materials into distinct target containers in front of different backgrounds. We verified that our proposed method has advantages in terms of high **success rate**, **data efficiency** and **generalizability**. With our progressive learning method, the success rate for the robotic pouring task can reach 77.5%, while the performance of the fine learning model can maintain even if we only use 25% of data for training. Moreover, with imaginary learning phase, the robot can increase the success rate for task execution in new scenarios from 29.1% to 79.2%.

To the best of our knowledge, this is the first time that one-shot domain adaptation has been carried out for a robotic pouring task in a physical environment. Moreover, we proposed the progressive learning method, which can be widely applied to different learning-based robotic manipulation tasks.

Appendix

Suppose that n is the total number of samples used for calculating the loss function, the adversarial loss on the observation samples in domain $\boldsymbol{R}$ can be calculated as follows:

$$\mathcal{L}_{\text{adv}}(\boldsymbol{G}, \boldsymbol{D_R}, \boldsymbol{H}, \boldsymbol{R}) = \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{D_R}\left(x_i^r\right) - 1\right)^2 + \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{D_R}\left(\boldsymbol{G}\left(x_i^h\right)\right)\right)^2 \tag{4}$$

Similarly, the adversarial loss on the observation samples in domain $\boldsymbol{H}$ can be calculated as follows:

$$\mathcal{L}_{\text{adv}}\left(\boldsymbol{G'}, \boldsymbol{D_H}, \boldsymbol{R}, \boldsymbol{H}\right) = \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{D_H}\left(\boldsymbol{G'}\left(x_i^r\right)\right)\right)^2 + \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{D_H}\left(x_i^h\right) - 1\right)^2 \tag{5}$$

The cycle consistency loss can be calculated as follows.

$$\begin{aligned}\mathcal{L}_{\text{cyc}}\left(\boldsymbol{G}, \boldsymbol{G}'\right) &= \mathcal{L}_{\text{cyc}}^{1} + \mathcal{L}_{\text{cyc}}^{2} \\ &= \frac{1}{n}\sum_{i=1}^{n}\left[\|\boldsymbol{G}'(\boldsymbol{G}(x_i^h)) - x_i^h\|_1\right] \\ &\quad + \frac{1}{n}\sum_{i=1}^{n}\left[\|\boldsymbol{G}(\boldsymbol{G}'(x_i^r)) - x_i^r\|_1\right] \end{aligned} \tag{6}$$

where ||.|| represents the L1 norm (Manhattan norm). The overall loss is computed by adding the adversarial loss of $\boldsymbol{G}(.)$ and $\boldsymbol{G}'(.)$ as well as the cycle consistency loss, which is defined as follows:

$$\begin{aligned}\mathcal{L}\left(\boldsymbol{G}, \boldsymbol{G}', \boldsymbol{D_H}, \boldsymbol{D_R}\right) &= \mathcal{L}_{\text{adv}}(\boldsymbol{G}, \boldsymbol{D_R}, \boldsymbol{H}, \boldsymbol{R}) \\ &\quad + \mathcal{L}_{\text{adv}}\left(\boldsymbol{G}', \boldsymbol{D_H}, \boldsymbol{R}, \boldsymbol{H}\right) \\ &\quad + \lambda\mathcal{L}_{\text{cyc}}(\boldsymbol{G}, \boldsymbol{G}') \end{aligned} \tag{7}$$

where λ is a parameter that controls the relative importance between the adversarial loss and the cycle consistency loss. The target is to solve a min-max problem as follows:

$$G^*, G'^* = \arg\min_{G,G'}\max_{D_H,D_R}\mathcal{L}\left(G, G', D_H, D_R\right) \tag{8}$$

REFERENCES

[1] Y. Wang, Y. Jiao, R. Xiong, H. Yu, J. Zhang, and Y. Liu, "MASD: A multimodal assembly skill decoding system for robot programming by demonstration," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 4, pp. 1722–1734, Oct. 2018.

[2] X. Fu, Y. Liu, and Z. Wang, "Active learning-based grasp for accurate industrial manipulation," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1610–1618, Oct. 2019.

[3] J. Chen et al., "Supervised semi-autonomous control for surgical robot based on Banoian optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 2943–2949.

[4] X. Liu, P. Huang, and Z. Liu, "A novel contact state estimation method for robot manipulation skill learning via environment dynamics and constraints modeling," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 3903–3913, Oct. 2022.

[5] D. Zhang et al., "Human–robot shared control for surgical robot based on context-aware sim-to-real adaptation," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 7694–7700.

[6] Y. Zhou, Y. Aytar, and K. Bousmalis, "Manipulator-independent representations for visual imitation," 2021, *arXiv:2103.09016*.

[7] T. Zhang et al., "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 1–8.

[8] Y. Ma, Y. Xie, W. Zhu, and S. Liu, "An efficient robot precision assembly skill learning framework based on several demonstrations," *IEEE Trans. Autom. Sci. Eng.*, early access, Jan. 31, 2022, doi: 10.1109/TASE.2022.3144282.

[9] G. Cheng, K. Ramirez-Amaro, M. Beetz, and Y. Kuniyoshi, "Purposive learning: Robot reasoning about the meanings of human activities," *Sci. Robot.*, vol. 4, no. 26, Jan. 2019, Art. no. eaav1530.

[10] P. M. L. A. Van Den Bemt, J. C. Idzinga, H. Robertz, D. G. Kormelink, and N. Pels, "Medication administration errors in nursing Homes using an automated medication dispensing system," *J. Amer. Med. Inform. Assoc.*, vol. 16, no. 4, pp. 486–492, Jul. 2009.

[11] N. Saigal, S. Baboota, A. Ahuja, and J. Ali, "Fast-dissolving intra-oral drug delivery systems," *Exp. Opinion Therapeutic Patents*, vol. 18, no. 7, pp. 769–781, Jul. 2008.

[12] M. Kennedy, K. Schmeckpeper, D. Thakur, C. Jiang, V. Kumar, and K. Daniilidis, "Autonomous precision pouring from unknown containers," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2317–2324, Jul. 2019.

[13] Z. Pan and D. Manocha, "Feedback motion planning for liquid pouring using supervised learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 1252–1259.

[14] A. Neumann et al., "'Kognichef': A cognitive cooking assistant," *KI-Künstliche Intelligenz*, vol. 31, no. 3, pp. 273–281, 2017.

[15] D. Zhang, Q. Li, Y. Zheng, L. Wei, D. Zhang, and Z. Zhang, "Explainable hierarchical imitation learning for robotic drink pouring," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 3871–3887, Oct. 2022.

[16] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," 2018, *arXiv:1811.06711*.

[17] W. Wang, R. Li, Y. Chen, Z. M. Diekel, and Y. Jia, "Facilitating human–robot collaborative tasks by teaching-learning-collaboration from human demonstrations," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 640–653, Apr. 2018.

[18] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.

[19] B. Zheng, S. Verma, J. Zhou, I. Tsang, and F. Chen, "Imitation learning: Progress, taxonomies and challenges," 2021, *arXiv:2106.12177*.

[20] B. Fang, S. Jia, D. Guo, M. Xu, S. Wen, and F. Sun, "Survey of imitation learning for robotic manipulation," *Int. J. Intell. Robot. Appl.*, vol. 3, no. 4, pp. 362–369, 2019.

[21] H. Guan, L. Wang, and M. Liu, "Multi-source domain adaptation via optimal transport for brain dementia identification," in *Proc. Uncertainty Artif. Intell.*, Apr. 2021, pp. 225–235.

[22] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–34, 2020.

[23] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 3630–3638.

[24] G. Koch et al., "Siamese neural networks for one-shot image recognition," in *Proc. ICML Deep Learn. Workshop*, vol. 2, Lille, France, 2015, pp. 1–30.

[25] D. Rezende et al., "One-shot generalization in deep generative models," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1521–1529.

[26] Y. Duan et al., "One-shot imitation learning," 2017, *arXiv:1703.07326*.

[27] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," in *Proc. Conf. Robot Learn.*, 2017, pp. 357–368.

[28] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[29] K. Bousmalis et al., "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4243–4250.

[30] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 7167–7176.

[31] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 597–613.

[32] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3722–3731.

[33] J. D. Langsfeld, K. N. Kaipa, R. J. Gentili, J. A. Reggia, and S. K. Gupta, "Incorporating failure-to-success transitions in imitation learning for a dynamic pouring task," in *Proc. Workshop Compliant Manipulation, Challenges Control*, Chicago, IL, USA, 2014, p. 4.

[34] T. Chen, Y. Huang, and Y. Sun, "Accurate pouring using model predictive control enabled by recurrent neural network," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 7688–7694.

[35] J. Pari, N. M. Shafiullah, S. P. Arunachalam, and L. Pinto, "The surprising effectiveness of representation learning for visual imitation," 2021, *arXiv:2112.01511*.

[36] D. Theckedath and R. R. Sedamkar, "Detecting affect states using VGG16, ResNet50 and SE-ResNet50 networks," *Social Netw. Comput. Sci.*, vol. 1, no. 2, pp. 1–7, Mar. 2020.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[38] O. Khatib, "Inertial properties in robotic manipulation: An object-level framework," *Int. J. Robot. Res.*, vol. 14, no. 1, pp. 19–36, Feb. 1995.

[39] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[40] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[41] C. Schenck and D. Fox, "Visual closed-loop control for pouring liquids," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 2629–2636.
[42] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1125–1134.
[43] A. Church et al., "Tactile sim-to-real policy transfer via real-to-sim image translation," in *Proc. Conf. Robot Learn.*, 2022, pp. 1645–1654.
[44] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2223–2232.
[45] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1857–1865.
[46] Z. Yi, H. Zhang, P. Tan, and M. Gong, "DualGAN: Unsupervised dual learning for image-to-image translation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2849–2857.
[47] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
[48] P. Sharma, L. Mohan, L. Pinto, and A. Gupta, "Multiple interactions made easy (MIME): Large scale demonstrations data for imitation," in *Proc. Conf. Robot Learn.*, 2018, pp. 906–915.
[49] L. Smith, N. Dhawan, M. Zhang, P. Abbeel, and S. Levine, "AVID: Learning multi-stage tasks via pixel-level translation of human videos," 2019, *arXiv:1912.04443*.
[50] M. Zhai, L. Chen, F. Tung, J. He, M. Nawhal, and G. Mori, "Lifelong GAN: Continual learning for conditional image generation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2759–2768.

Dandan Zhang (Member, IEEE) received the Ph.D. degree from the Computing Department, Imperial College London, in 2021. She is currently an Assistant Professor with the Department of Engineering Mathematics, University of Bristol, affiliated with the Bristol Robotics Laboratory. She is also an Honorable Researcher at the Department of Electrical and Electronic Engineering, Imperial College London. Since she joined the University of Bristol, she has been working with the Dexterous Robotics Group. She has cross-disciplinary interests in robotics and AI. She targets enhancing the level of autonomy for multiscale robotics systems. The ultimate goal is to develop next-generation robots empowered by artificial general intelligence with super-human capabilities. She envisions that intelligent robots will reshape our world by providing tangible benefits in our daily life, contributing to the healthcare systems.

Wen Fan (Graduate Student Member, IEEE) received the B.Sc. degree in automation from the Hefei University of Technology, China, in 2020, and the M.Sc. degree in advanced control and system engineering from The University of Manchester, U.K., in 2021. He is currently pursuing the Ph.D. degree with the Intelligent Robotics Group, Department of Engineering Mathematics, University of Bristol, U.K. His research interests include tactile robotics, robot learning, and XAI.

John Lloyd received the B.Sc. degree in physics with electronics and the Ph.D. degree in computer science from The University of Manchester, Manchester, U.K. He is currently a Senior Research Associate in robotics and AI with the University of Bristol, Bristol, U.K., working on the application of machine learning to robot touch. His research interests include machine learning, artificial intelligence, robotics, and tactile sensors.

Chenguang Yang (Senior Member, IEEE) received the Ph.D. degree in control engineering from the National University of Singapore, Singapore, in 2010. He completed his post-doctoral training in human–robotics from the Imperial College London, London, U.K. His research interests include human–robot interaction and intelligent system design. He is a fellow of the Institute of Engineering and Technology (IET) and the British Computer Society (BCS). He was awarded the U.K. EPSRC UKRI Innovation Fellowship and the individual EU Marie Curie International Incoming Fellowship. As the lead author, he won the IEEE TRANSACTIONS ON ROBOTICS Best Paper Award in 2012 and the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS Outstanding Paper Award in 2022. He is the Corresponding Co-Chair of IEEE Technical Committee on Collaborative Automation for Flexible Manufacturing.

Nathan F. Lepora (Member, IEEE) received the B.A. degree in mathematics and the Ph.D. degree in theoretical physics from the University of Cambridge, U.K. He is currently a Professor of Robotics and AI with the University of Bristol, U.K. He leads the Dexterous Robotics Group that researches human-like dexterity with soft tactile robots and AI, such as deep learning. He was a recipient of the Leverhulme Research Leadership Award on "A Biomimetic Forebrain for Robot Touch." His research group won the "Contributions in Soft Robotics Research" category in the 2016 Soft Robotics Competition. He co-edited the book *Living Machines* that won the 2019 BMA Medical Book Awards basic and clinical sciences category.