



O₂A: One-Shot Observational Learning with Action Vectors

Leo Pauly^{1*}, Wisdom C. Agboh¹, David C. Hogg¹ and Raul Fuentes²

¹University of Leeds, Leeds, United Kingdom, ²RWTH Aachen University, Aachen, Germany

We present O₂A, a novel method for learning to perform robotic manipulation tasks from a single (one-shot) third-person demonstration video. To our knowledge, it is the first time this has been done for a single demonstration. The key novelty lies in pre-training a feature extractor for creating a perceptual representation for actions that we call “action vectors”. The action vectors are extracted using a 3D-CNN model pre-trained as an action classifier on a generic action dataset. The distance between the action vectors from the observed third-person demonstration and trial robot executions is used as a reward for reinforcement learning of the demonstrated task. We report on experiments in simulation and on a real robot, with changes in viewpoint of observation, properties of the objects involved, scene background and morphology of the manipulator between the demonstration and the learning domains. O₂A outperforms baseline approaches under different domain shifts and has comparable performance with an Oracle (that uses an ideal reward function). Videos of the results, including demonstrations, can be found in our: project-website.

OPEN ACCESS

Edited by:

Christophoros Nikou,
University of Ioannina, Greece

Reviewed by:

Juan Wachs,
Purdue University, United States
Jose Delpiano,
University of the Andes, Chile, Chile

*Correspondence:

Leo Pauly
cnp@leeds.ac.uk

Specialty section:

This article was submitted to
Robot and Machine Vision,
a section of the journal
Frontiers in Robotics and AI

Received: 26 March 2021

Accepted: 13 July 2021

Published: 02 August 2021

Citation:

Pauly L, Agboh WC, Hogg DC and
Fuentes R (2021) O₂A: One-Shot
Observational Learning with
Action Vectors.
Front. Robot. AI 8:686368.
doi: 10.3389/frobt.2021.686368

Keywords: observational learning, visual perception, reinforcement learning, transfer learning, robotic manipulation

1 INTRODUCTION

Learning new manipulation tasks has always been challenging for robotic systems, whether it is a simple mobile manipulator or a complex humanoid robot. Programming manually step by step (Finkel et al., 1975) is one of the earlier solutions to this problem. But this is labour intensive, requires specialist expertise and lacks autonomy. It is therefore not suitable for consumer robots and fully autonomous systems. Learning from Demonstrations (LfD) (Atkeson and Schaal, 1997) is a potential solution to this problem. It requires only demonstrations of the task for the robot to learn from. Even though LfD has been studied widely, most previous works have stayed within the “Imitation Learning” (Duan et al., 2017; Pathak et al., 2018; Peng et al., 2018; Argall et al., 2009) paradigm, where demonstrations are made from an egocentric perspective, either visually or kinesthetically. This requires the inconvenience of kinaesthetic guidance or teleoperation and also the rich source of third-person demonstrations available on the internet cannot be used. Therefore, in this paper we study the problem of LfD under the “Observational Learning” (Bandura 1986; Pauly, 2018; Torabi et al., 2019; Borsa et al., 2019) paradigm, where the demonstrations are observed as a third-person. This introduces the key challenge in observational learning, the shift between the demonstration and the learning domains. The domain shift can arise due to changes in viewpoints of observation, properties of objects used, scene background or morphology of the manipulator performing the task.

In this paper we present O₂A (One-shot Observational learning with Action vectors), for one-shot observational learning of robotic manipulation tasks under different domain shifts. One-shot learning here means that only a single demonstration of the new task is required for learning. (Note that, it does not refer to the number of trial and error executions by the robot during learning

from that single demonstration). We use an abstract perceptual representation: the “action vector”, which is the task-discriminative and domain-invariant representation of the action in a video. The action vector is extracted using a 3D-CNN (Tran et al., 2015), pre-trained on a generic action dataset as an action classifier (we use UCF101 (Soomro et al., 2012) as the pre-training dataset for our experiments). Through our evaluation on a new “Leeds Manipulation Dataset” (LMD), we show that the pre-trained action vector extractor can generalise to unseen manipulation tasks. The action vectors from the demonstration and trial robot execution video clips are then compared to generate a reward for the reinforcement learning algorithm. The algorithm learns an optimal control policy that performs the demonstrated task. Our experiments in simulation (with reaching, pushing tasks) and on a real robot (with pushing, hammering, sweeping, striking tasks) show that O₂A can perform well under different domain shifts. Our contributions can be summarised as follows:

- Implementing for the first time, to the best of our knowledge, a method for observational learning of robotic manipulation tasks from a single demonstration.
- O₂A can handle shifts between the demonstration and the learning domains, caused by changes in viewpoint of observation, object properties, morphology of the manipulator and scene background.
- And finally, we pre-train the action vector extractor on a generic action dataset instead of task-specific manipulation videos. The extractor generalises to unseen manipulation tasks by learning the shared underlying visual dynamics.

Upcoming sections are arranged as follows: **Section 2** discusses related works, **Section 3** formulates the problem and describes the proposed method, **Sections 4** and **5** report on experiments conducted and finally **Section 6** presents the conclusions.

2 RELATED WORK

Observational learning: Origins of observational learning of robotic manipulation tasks can be traced back to works from the 1990s (Suehiro, 1994; Kuniyoshi et al., 1994; Bakker and Kuniyoshi, 1996). Most of the early methods required assistance in observing the demonstrations. This assistance was provided by motion capture systems (Ijspeert et al., 2001; Ijspeert et al., 2002; Field et al., 2009), visual detectors (Ramirez-Amaro et al., 2017; Sieb et al., 2020; Zhang and Nikolaidis, 2019), skeleton tracking (Cabrera and Wachs, 2017), trackers/markers (Dillmann, 2004; Dragan and Srinivasa, 2012; Gupta et al., 2016) or a combination of the above (Kuniyoshi et al., 1994). However, the entities to be tracked or detected must be known beforehand and only demonstrations using these entities can be learned.

With the advent of deep learning (LeCun et al., 2015; Goodfellow et al., 2016), it was possible to learn visual features characterising the task directly from raw RGB videos. The features are extracted from raw videos using a variety of

methods: deep metric learning (Sermanet et al., 2018), generative adversarial learning (Stadie et al., 2017), domain translation (Liu et al., 2018; Smith et al., 2019; Sharma et al., 2019), transfer learning (Sharma et al., 2018; Sermanet et al., 2017), action primitives (Jia et al., 2020), predictive modelling (Tow et al., 2017), video to text translation (Yang et al., 2019), meta-learning and (Yu et al., 2018a; Yu et al., 2018b). A comparison of these methods is given in the **Table 1** and a detailed study can be found in (Pauly, 2021).

A limitation of these methods is the requirement of a large number of demonstrations for learning new tasks: the feature extractor is trained separately for each of the new task to be learned. Hence demonstration videos are to be collected in substantial numbers for each task. In contrast, our method requires only a single demonstration (hence one-shot) to learn a new task, since pre-trained feature extractors are used. A second limitation is the constrained domain shifts: In existing approaches, assumptions are made regarding shift between learning and demonstration domains. For example viewpoint of observation is fixed (Sermanet et al., 2017) or manipulators with similar morphologies (Liu et al., 2018) are used. Our method O₂A, does not make any such assumptions and can learn under unconstrained domain shifts.

Pre-training with large generic datasets: Pre-training on large generic datasets has become common in the fields of computer vision and natural language processing. Models are first pre-trained on a large generic dataset(s) in a supervised or unsupervised manner. After pre-training, the models are used to solve downstream tasks with minimum/no fine-tuning. Generic language models such as ELMo (Peters et al., 2018), GPT (Radford et al., 2018; Radford et al., 2019; Brown et al., 2020), BERT (Devlin et al., 2018) have shown success in solving several downstream language processing tasks. Similarly, ImageNet models (Xie and Richmond, 2018), Image-GPT (Chen et al., 2020), BiT models (Kolesnikov et al., 2019) have demonstrated that this approach can be applied for computer vision problems as well. We introduce a similar concept into visual robotic manipulation. The action vector extractor is pre-trained using a large generic action dataset and then generalised to manipulation tasks for observational learning.

3 PROPOSED METHOD

3.1 Action Vectors

Action vectors are the core of the O₂A method. An action vector is the abstract task-discriminative and domain-invariant perceptual representation of the action being carried out in a video. In O₂A, the action vector extraction is based on the following two assumptions, which we validate in **Section 4**.

- 1) The spatio-temporal features generated by the final layers of an action classifier pre-trained on a generic action dataset, are task-discriminative domain-invariant. The features from the videos depicting similar actions should be identical irrespective of the domain in which they are recorded. The assumption is reasonable since the action classifier makes use

TABLE 1 | Observational learning methods in existing literature are compared. O₂A requires only a single demonstration to learn new tasks. It does not use any robot data for training the action vector extractor and also works well under different domain shifts.

References	No: of video demonstrations required per task (including to train the feature extractor/s)	Is robot data required for training the feature extractor/s ?	Viewpoint invariant ?	Invariant to changes of object properties ?	Invariant to changes in scene background ?	Invariant to changes of morphology of the manipulator ?
Sermanet et al. (2018)	~40 min of human demonstrations + ~20 min random robot manipulation data	✓	✓	✓	✓	✓
Stadie et al. (2017)	An expert policy is used instead of direct demonstrations	✓	✓	✓	✓	✓
Liu et al. (2018)	~60–3,000 human demonstrations using additional tools	✗	✓	✓	✓	✗
Smith et al. (2019)	~20–30 human demonstrations + ~300–500 random human and robot images	✓	✗	NA	NA	✓
Sharma et al. (2019)	~230 human demonstrations + corresponding robotic joint angle data	✓	✓	✓	NA	✓
Sharma et al. (2018)	~200–400 human demonstrations + corresponding robotic joint angle data	✓	✓	✓	NA	✓
Sermanet et al. (2017)	~12 human demonstrations	✗	✗	✓	NA	✓
Jia et al. (2020)	~50–100 human demonstrations	✗	✓	✓	✓	✓
Tow et al. (2017)	Uses both human and robot task demonstrations (exact numbers unknown)	✓	✓	NA	NA	✓
Yang et al. (2019)	~2,990 human demonstrations	✗	✓	✓	✓	✓
Yu et al. (2018b)	1 (but uses closely related supplementary task demonstrations. Requires ~600–1,200 robot and ~600–1,200 human demonstrations per task)	✓	✓	✓	✓	✓
Yu et al. (2018a)	1 (but requires large number of action primitive demonstrations. ~600–1,200 robot and ~600–1,200 human demonstrations per action primitive)	✓	NA	✓	✓	✓
O ₂ A	Only 1 demonstration (human demonstration with or without using additional tools)	✗	✓	✓	✓	✓

of the same layer outputs to identify actions, independently of different camera angles, varying scene backgrounds, illumination conditions, actors/manipulators, object appearances, interactions, pose and scale.

- 2) The action vector extraction model pre-trained on a generic dataset can generalise to unseen manipulation tasks used in robotic observational learning. The intuition is that the underlying visual dynamics between generic action datasets and manipulation tasks are the same. For example, it is the same physical laws of dynamics governing object interactions, both for a cricket shot as well as a robot striking cubes.

3.1.1 Network Architecture and Dataset

Our 3D-CNN model consists of eight 3D convolutional layers, five 3D maxpooling layers and three fully connected layers. The ReLU (Nair and Hinton, 2010) activation function is used for all the convolutional and fully connected layers except the final layer, where we use a Softmax function. The layer wise network architecture along with the kernel sizes, input and output dimensions are given in the **Supplementary Section S1**. We use the UCF101 action dataset as the generic dataset for our experiments. It consists of 13320 real world action videos from YouTube each lasting around 7 s on average, classified into 101

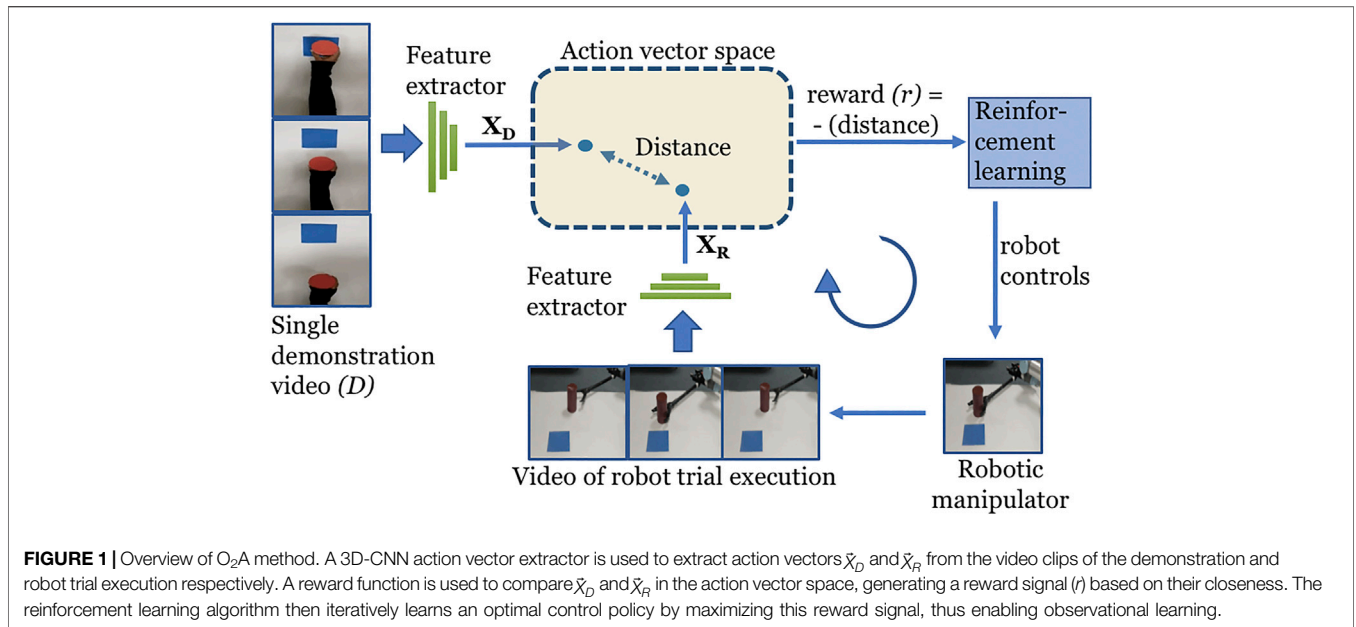
action categories. The dataset has a large diversity both in terms of variety of actions and domain settings within the same class videos. A dropout of 0.5 is used for the fully connected layers during training to avoid overfitting. We also use a zero-padding layer between the last convolution and pooling layers to control the shrinkage of dimensions.

3.1.2 Pre-training Action Vector Extractors

For pre-training, we first uniformly downsample UCF101 videos in time into 16 frames for providing a fixed-length representation for each video clip. We also resize videos into 112×112 pixels to standardize the size. We apply the same pre-processing steps to videos of demonstrations and robot trial executions for action vector extraction during observational learning. These downsampled and resized videos are then used for pre-training the model for action classification from scratch. The pre-training objective is the cross-entropy loss function given as:

$$L(y, \bar{y}) = - \sum_{i=0}^{NC-1} y_i \log(\bar{y}_i) \quad (1)$$

where, \bar{y}_i is the i th value in the predicted output and y_i is the corresponding one-hot encoded ground truth value, both of



which are expressed as probabilities. NC is the number of classes and for UCF101 dataset $NC = 101$. The training details are given in the **Supplementary Section S1**. The pre-trained model will be referred to as “NN:UCF101” hereafter.

After training, we use features from one of the final layers of NN:UCF101 as the action vector. Our experiment (reported in **Section 4.1**) shows that the features from layers pool5 (size: 8,192) and fc6 (size: 4,096) are best suited to be used as the action vector. We report results, both when the features from pool5 and fc6 layers are used as the action vector in this paper.

3.2 One-Shot Observational Learning

The overview of O₂A is shown in **Figure 1**. The robot views both the demonstration and its own trial executions from a camera mounted in a fixed position above the manipulator. With reference to **Figure 1**, let D be the single demonstration video clip of a task to be learned. We extract the n -dimensional action vectors \vec{x}_D and \vec{x}_R from the demonstration video D and the video clip of a trial robot execution respectively. The reward (r) for the reinforcement learning is then calculated as the negative of the euclidean distance between action vectors \vec{x}_D and \vec{x}_R as given below:

$$r = -\|\vec{x}_D - \vec{x}_R\|_2 \tag{2}$$

Thus the reward directly measures the closeness of the actions in the demonstration and of the robot trial execution. The reinforcement learning will then maximize this reward function to learn an optimal control policy. This optimal control policy will enable the robotic manipulator to perform the demonstrated task.

3.2.1 Reinforcement Learning of the Task

Any reinforcement learning algorithm can be used with our method. In the simulation experiment, we use the Deep

Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2015) to estimate the optimal control policy. The states used by the control policy are instantaneous visual observations of the environment (as observed by the robotic system). We make use of a VGGNet pre-trained on ImageNet (Simonyan and Zisserman, 2015) for converting raw RGB images into visual state features. The 4,608 long feature obtained from the last convolutional layer of the VGG-16 network is used as the instantaneous state representation.

Reinforcement learning in real robots is an active area of research and remains a challenging problem. So we use a manipulation planning algorithm, the Stochastic Trajectory Optimisation (STO) (Agboh and Dogar, 2018; Kalakrishnan et al., 2011), for the real robot experiment. Using a different manipulation algorithm does not undermine the effectiveness of our method. The objective behind the experiment is to show that O₂A reward can successfully guide a robotic manipulation algorithm to learn the demonstrated tasks. STO generates an optimal control sequence by iteratively improving on the previous sequence guided by our reward function. The cost function C , to be minimized is calculated as:

$$C = r^2 \tag{3}$$

Additionally, this shows that our method is agnostic to the robotic manipulation algorithms and can directly be used with more advanced algorithms when available in future.

4 ACTION VECTOR ANALYSIS

In this section, we aim to validate our assumptions for the proposed action vector extraction method explained in **Section 3.1**. First we collect a manipulation dataset, the “Leeds

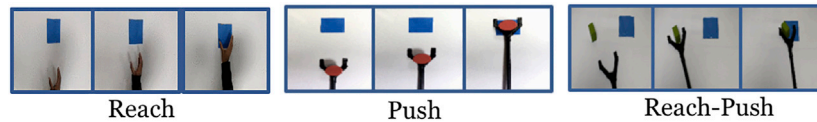


FIGURE 2 | Snapshots of sample videos from LMD. Identical looking task classes are used to make the task differentiation more challenging.

Manipulation Dataset (LMD)”. Note that this dataset is only used for evaluation and not used during training of the action vector extractor. LMD consists of videos of three different manipulation tasks: reach, push and reach-push, examples of which are shown in **Figure 2**. The task videos are collected directly with a human hand and by using tools resembling robotic manipulators/end effectors. Each class consists of 17 videos with variations in viewpoint, object properties, scene background and morphology of manipulator within each class. Note that, identical looking task classes were carefully selected and same set of objects and manipulators were used across tasks for collecting videos. These choices are deliberate to make the task differentiation more challenging. Under these circumstances, only an efficient action vector extractor can produce task-discriminative and domain-invariant action vectors for different task classes in LMD.

4.1 Clustering Analysis

We conduct the experiment to identify which one of the final layers of NN:UCF101 provides the best action vector for manipulation tasks. We use the quality of the clusters in the action vector space, as a measure to understand how task-discriminative and domain-invariant are the action vectors from different layers of NN:UCF101 model. The more the action vectors are task-discriminative and domain-invariant, the better the clustering of the action vectors from the same class will be. To analyse the quality of the clusters, we use a standard clustering evaluation measure, the ARI (Hubert and Arabie, 1985) score. The ARI score measures the extent to which the predicted clustering corresponds to the and ground truth clusters by counting pairs that are assigned in the same or different clusters. ARI values are bounded by $[-1, 1]$, where -1 is the lowest score, 0 indicates random clustering and 1 shows that the predicted clustering corresponds to the ground truth clusters perfectly.

For the experiment, we extract the action vector from the pool5, fc6, fc7 and fc8 layers of the NN:UCF101 model, for all the 17 videos in LMD. The Baseline-R is obtained using features from the pool5 layer of the same NN:UCF101 model but initialised with random weights. The features extracted from each layer are then clustered using the K-means clustering algorithm. The value of $K = 3$ is used, corresponding to the number of task classes. After clustering, the predicted cluster labels are evaluated against ground truth labels and ARI scores are calculated. The results of the experiment are tabulated in **Table 2**.

The ARI value for Baseline-R is close to zero as expected and gives us the baseline to compare with. The ARI score increases when features from pool5 to fc6 layers are used as the action

TABLE 2 | ARI scores. Results show that the features from layer pool5 and fc6 of the NN:UCF101 model are best suited to be used as action vectors.

Layer	ARI score
Baseline-R (random weights)	0.07
pool5	0.26
fc6	0.34
fc7	0.19
fc8	0.14

Best results presented in bold.

vector, but drops for the final fc7 and fc8 layers. The results indicate that the features from pool5 and fc6 layers of the NN:UCF101 model are the most suitable to be used as the action vector. These results are in agreement with the previous works (Azizpour et al., 2015; Athiwaratkun and Kang, 2015) that study transferability of features from different layers of a pre-trained CNN to new downstream problems. Specifically, Azizpour et al. (2015) have shown that the first fully connected layer after the convolutional layers of a pre-trained (for classification) network produced the most generic features for a range of 15 downstream problems. An experiment (reported in **Supplementary Section S3**) using optimal K values for each layer also shows that the features from pool5 and fc6 layers are best suited to be used as the action vector. The optimal K value is obtained by performing clustering analysis and calculating ARI scores for each value of K from 1 to 51 (the total number of samples in LMD). Furthermore, we also performed clustering analysis (while not reported here) when features from different layers are concatenated and used as the action vector. Concatenating features did not produce any significant improvements in the performance.

4.2 Class Similarity Scores

Here we calculate the interclass and intraclass similarity scores for different classes of LMD in the action vector space. For that, we extract action vectors from pool5 and fc6 layers of the NN:UCF101 model, for all the 51 videos in LMD. The Baseline-R is obtained using features from pool5 layer of the same NN:UCF101 model but initialised with random weights. The similarity score between a pair of action vectors, is shown as the cosine of the angle between them. The similarity scores are bounded by $[-1, 1]$ with -1 indicating diametrically opposite vectors and 1 indicating coinciding vectors.

The results are tabulated in **Table 3**. For each chosen feature layer, the diagonal values represent the average of similarity scores between pairs of action vectors from the same class. And the non-

TABLE 3 | Class similarity scores. The intraclass similarity (diagonal values) are greater than the rest of the values, indicating adequate task-discrimination and domain-invariance.

	Reach	Push	Reach-push
Baseline-R (random weights)			
Reach	0.9873	0.9870	0.9870
Push	0.9870	0.9874	0.9868
Reach-push	0.9870	0.9868	0.9889
NN:UCF101 (pool5)			
Reach	0.7391	0.7371	0.6897
Push	0.7371	0.7547	0.6852
Reach-push	0.6897	0.6852	0.7578
NN:UCF101 (fc6)			
Reach	0.4994	0.5001	0.4052
Push	0.5001	0.5352	0.4022
Reach-push	0.4052	0.4022	0.4978

Best results presented in bold.

diagonal values are the average of similarity scores between pairs of action vectors from different classes. The diagonal values are greater than the rest of the values indicating adequate task-discrimination and domain-invariance for the action vectors extracted. The only exception is for layer fc6 where a greater inter-class similarity score is observed between reach and push classes than the intraclass similarity score for reach class. Provided that both tasks are extremely similar, these results are promising.

We also visualize these action vectors from LMD, projected into 2D using PCA, which are shown in **Figure 3**. The clustering of action vectors from the same classes, when compared to the Baseline-R is evident. This further indicates the task-discrimination and domain-invariance of our action vectors. It must be noted that this visualisation collapses the vectors, of much greater dimensions, into a 2D space, which might be causing some of the “artificial” overlaps.

The class similarity scores and visualization shows that our pre-trained action vector extractor can generalise to unseen manipulation tasks. In the next section we show how the action vector is used for observational learning and how well O_2A performs under different domain shifts.

5 ROBOTIC EXPERIMENTS

To explore the resilience of our method to shifts between the demonstrator and learner domains, we conducted the experiments with six different domain shifts, as defined in **Table 4**. The tasks used are reaching and pushing in simulation and pushing, hammering, sweeping and striking for the real robot experiment. The task definitions and completion measures are given in **Table 5**. Note that the task completion measures are only used for evaluating the performance of O_2A and not used during learning.

5.1 Simulation Experiment

We set up the simulation learning domain with a 3DOF robotic manipulator for reaching and pushing using OpenAI Gym (Brockman et al., 2016) and the MuJoCo physics engine (Todorov et al., 2012). In each setup (characterising a domain shift), we collect a single demonstration in the real world and run DDPG algorithm 10 times. Each run has 20 episodes per run and the number of steps per episode are 60 and 160 for reaching and pushing respectively. We use architectures similar to Lillicrap et al. (2015) for the actor and critic networks. The hyper-parameters used are given in the **Supplementary Section S2**. For each run, the DDPG returns a control policy that corresponds to the maximum reward obtained. After training, we pick the top-2 (Henderson et al., 2018) control policies with the highest rewards, and the task completion measures are calculated. The top control policies were selected to avoid policies from poorly performing runs affecting the overall performance. The output of the control policy are the robotic controls with a size of three corresponding to each of the joints. The robotic controls could be torques, joint angles or velocities of the manipulator. In our experiment we have used joint angles. We perform the experiment with action vectors extracted from both pool5 and fc6 layers of NN:UCF101 model. **Figure 4** shows snapshots of the demonstration and execution of the corresponding learned policy for selected setups. Videos of the simulation experiment results, including demonstrations are available in the project-page.

We compare our method with an oracle and two baseline approaches. The oracle is trained by using the corresponding task

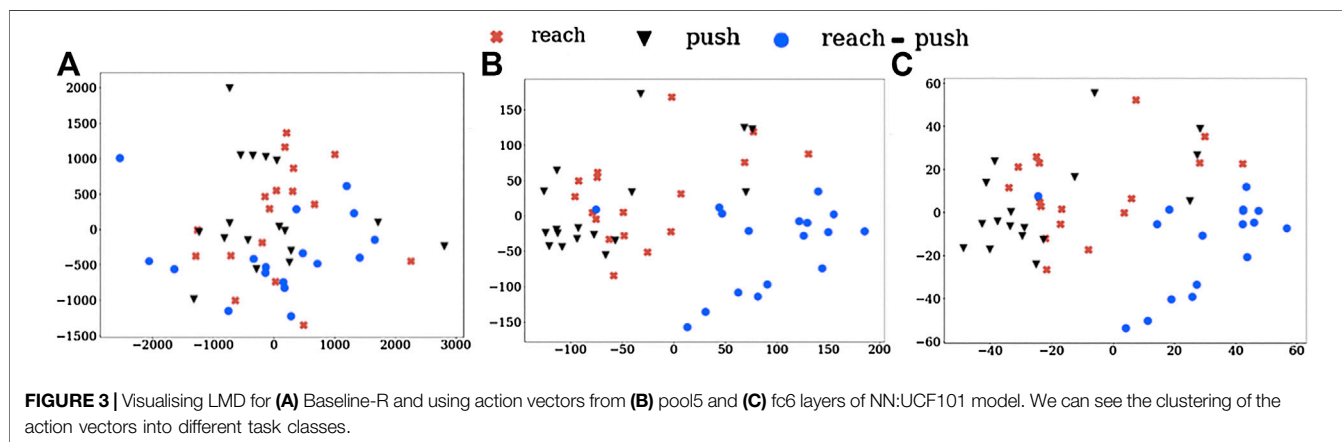


FIGURE 3 | Visualising LMD for (A) Baseline-R and using action vectors from (B) pool5 and (C) fc6 layers of NN:UCF101 model. We can see the clustering of the action vectors into different task classes.

TABLE 4 | Domain shifts used in our experiments.

	Domain shift
I	Observation viewpoint, object properties, morphology of the manipulator and scene background remain the same in the demonstration and learning domain
V	Observation viewpoint is different between the demonstration and the learning domain; other factors remain unchanged
Obj	Objects with different colour (for pushing, reaching and hammering tasks) or shape (hammering task) used in the learning domain
Obj+V	Both the viewpoint of observation and object properties vary between the demonstration and the learning domains
BG	Background clutter is introduced to the scene in learning domain, which was not present during the demonstration
M	Manipulators with different morphologies used in the demonstration and the learning domain. Demonstrations with a human hand (reaching and pushing tasks) and with a manipulator with a different morphology (hammering task) used

TABLE 5 | Task definitions and completion measures.

Task	Description	Task completion measure
Reaching (simulation)	Reach a target zone	1-(final distance/initial distance between the center of the manipulator and the center of the target zone)
Pushing (simulation and real robot)	Push an object into the target zone	1-(final distance/initial distance between the centers of the target zone and the pushed object)
Hammering (real robot)	Hammer the target object	1-(minimum distance/initial distance between the hammer and the object during the execution)
Sweeping (real robot)	Sweep crumpled cardboard pieces to the dustbin	The number of cardboard pieces in the dustbin after execution/total number of the cardboard pieces
Striking (real robot)	Strike down a block of cubes	1-(minimum distance/initial distance between the blocks and manipulator during execution)

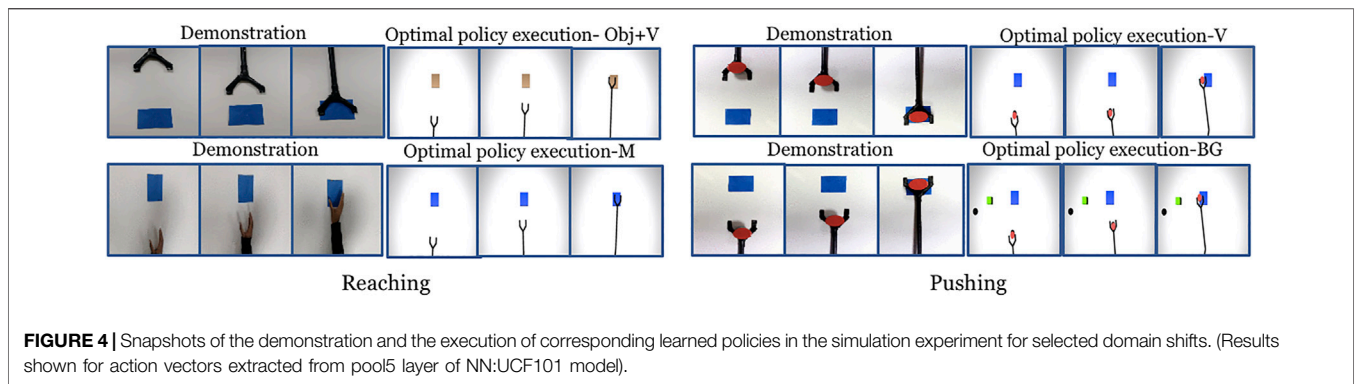


FIGURE 4 | Snapshots of the demonstration and the execution of corresponding learned policies in the simulation experiment for selected domain shifts. (Results shown for action vectors extracted from pool5 layer of NN:UCF101 model).

completion measure specified in **Table 5** as the reward, in place of a reward derived from action vectors. It represents the upper bound on performance. The two baselines represent a video clip by averaging a “static” representation for each frame, in contrast to the spatio-temporal representation used in O_2A . Rewards are then generated using these representations. In Baseline-1, features from the output of the last convolutional layer of the ImageNet (Simonyan and Zisserman, 2015) pre-trained VGG-16 network are used and in Baseline-2, HOG (Dalal and Triggs, 2005) features are used. The average of the task completion measures for the top two control policies for oracle, O_2A and the baseline approaches are plotted in **Figure 5**. The learned policies from O_2A were successful in performing the demonstrated task under different domain shifts with good task completion measures. It also significantly outperforms both baseline approaches and has a comparable performance to the oracle.

We further analysed the quality of the rewards generated in O_2A , the baseline approaches and the Oracle. To compare, we calculate the Pearson correlation coefficient (Benesty et al., 2009) between the episodic perceptual rewards (O_2A , baselines) and the Oracle rewards for the top two runs. A high positive correlation [typically >0.5 (Tipping and Bishop, 1999)] indicates that the perceptual rewards are as good as the Oracle rewards. All the results are tabulated in **Table 6**. From the results, the correlation coefficients are greater than 0.5 in all the cases for O_2A , indicating that our rewards are as accurate as the Oracle rewards. Also, the correlation is higher and positive compared to the baselines for a range of domain shifts showing the superior performance of our method.

5.1.1 Trajectory Maps

Here we plot the trajectories followed by the robotic manipulator in each episode during reinforcement learning of the task. This

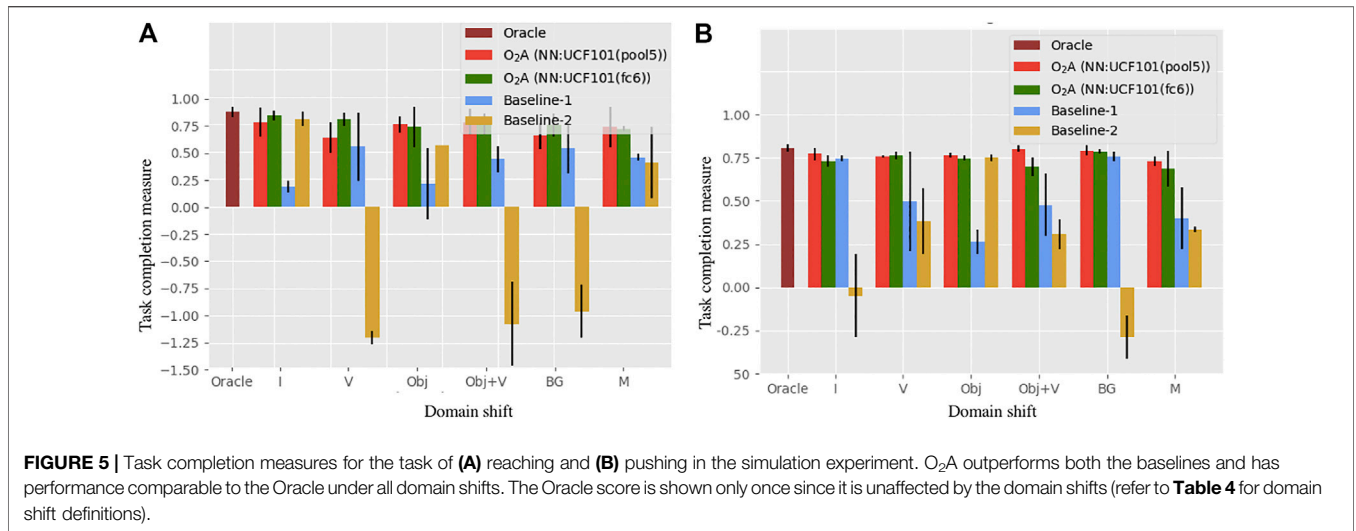


TABLE 6 | Pearson correlation coefficients between the rewards from the Oracle, and from O₂A and two baselines. The coefficients are generally highest and positive for O₂A rewards compared to baseline approaches.

	I	V	Obj	Obj+V	BG	M
Task 1: Reaching						
O ₂ A [NN:UCF101 (pool5)]	0.8567 ± 0.0079	0.7807 ± 0.0531	0.8209 ± 0.0157	0.6448 ± 0.2146	0.7736 ± 0.0007	0.9605 ± 0.0048
O ₂ A [NN:UCF101 (fc6)]	0.8318 ± 0.0600	0.7911 ± 0.0588	0.8199 ± 0.0718	0.8620 ± 0.0713	0.8108 ± 0.1126	0.8761 ± 0.0032
Baseline-1	0.5872 ± 0.1744	0.4069 ± 2,361	0.6112 ± 0.2612	0.6099 ± 0.901	0.5289 ± 0.0189	0.0487 ± 0.0448
Baseline-2	0.7387 ± 0.0681	-0.8106 ± 0.0086	0.7115 ± 0.1272	-0.8189 ± 0.0501	-0.5738 ± 0.0337	0.1256 ± 0.0629
Task 2: Pushing						
O ₂ A [NN:UCF101 (pool5)]	0.9345 ± 0.0034	0.9413 ± 0.0362	0.6943 ± 0.1419	0.8650 ± 0.0847	0.8552 ± 0.0677	0.6594 ± 0.1834
O ₂ A [NN:UCF101 (fc6)]	0.8037 ± 0.1125	0.8826 ± 0.0239	0.6898 ± 0.1927	0.8179 ± 0.0702	0.9147 ± 0.0099	0.8489 ± 0.0987
Baseline-1	0.9372 ± 0.0270	0.8908 ± 0.0615	0.5817 ± 0.3124	0.7488 ± 0.0631	0.8978 ± 0.0704	0.5797 ± 0.1141
Baseline-2	0.0173 ± 0.4550	-0.1346 ± 0.3410	0.5900 ± 0.1625	-0.4352 ± 0.1292	-0.5386 ± 0.1243	0.3700 ± 0.5195

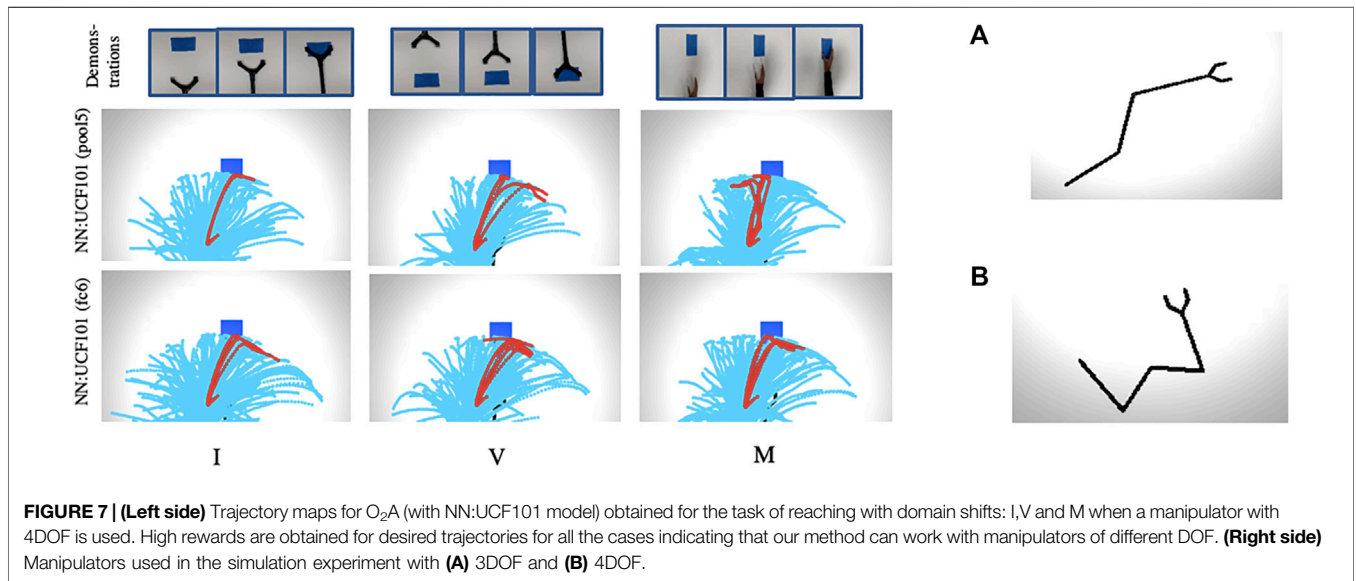
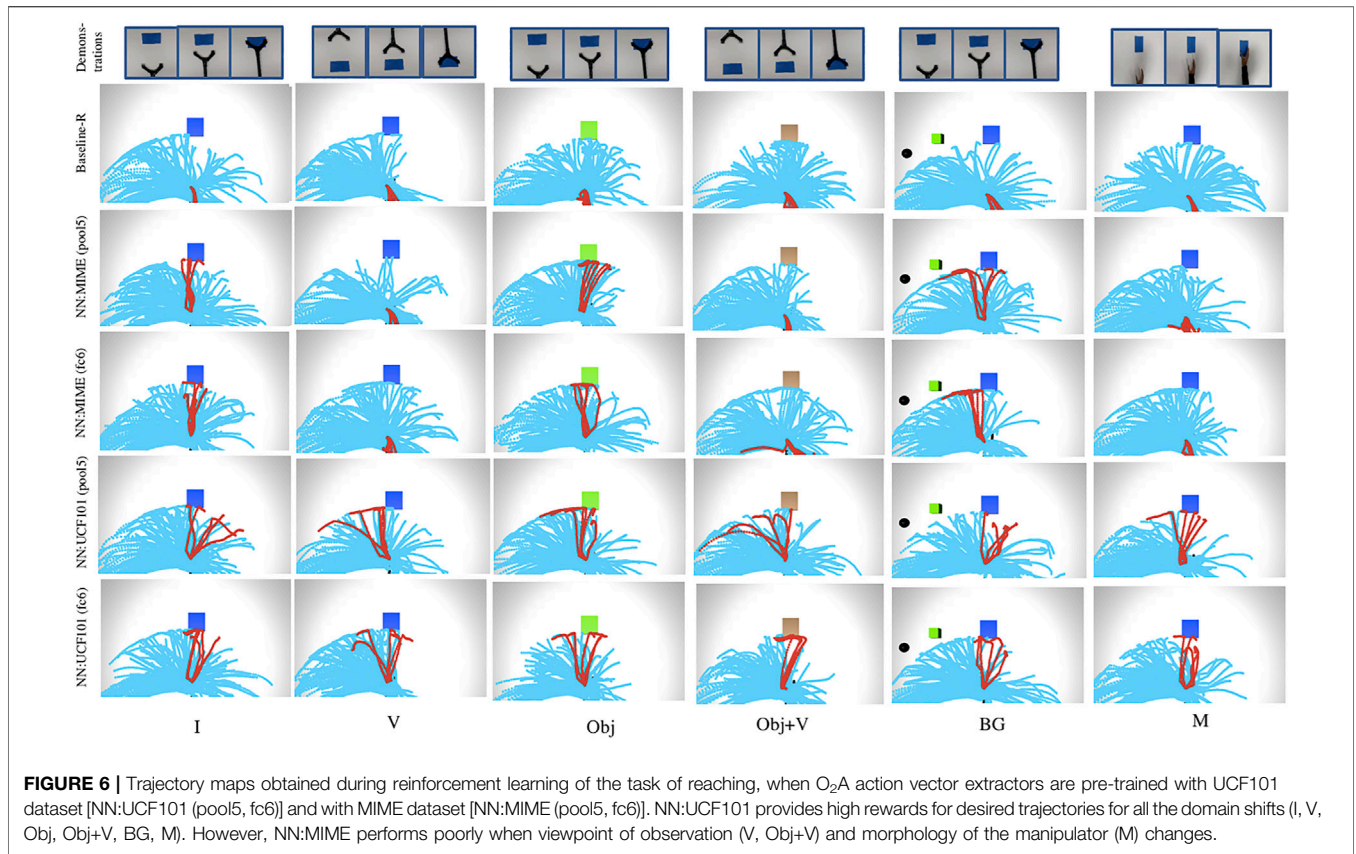
visualisation will help to understand, if high rewards are obtained for desired trajectories while learning the demonstrated task. The top-5 trajectories with the highest reward values obtained during task learning are coloured with red and the rest of the trajectories are in blue.

We also show the results when O₂A action vector extractors are pre-trained with a manipulation task dataset, the Multiple Interactions Made Easy (MIME) dataset (Sharma et al., 2018). MIME dataset consists of 8260 videos of 20 commonly seen robotic manipulation tasks, executed by a human as well as a Baxter robot. This model is referred to as ‘NN:MIME’. The aim is to study how well O₂A performs when pre-trained on a task specific manipulation dataset compared to a generic dataset. The results are shown in **Figure 6** (for reaching task) and in **Supplementary Section S4** (for pushing task). The reward values for desired trajectories in Baseline-R are low in all the cases as expected. The results indicate that, when NN:UCF101 is used, high rewards are generated for desired trajectories for all domain shifts. However NN:MIME performs poorly for changes in viewpoint and manipulator used. An insight into this is that, even though the MIME dataset consists of large

number of manipulation task examples, the variations in terms of viewpoints and manipulators used are limited. In contrast UCF101 contains examples with extensive range of variations in domain settings like viewpoint and manipulator morphology.

Further, we plotted trajectory maps for O₂A (with NN:UCF101 model) for the task of reaching when a 4DOF manipulator is used. The results are given in **Figure 7** for three different domain shifts: I, V and M, along with the manipulators used in the simulation experiment. The results show that O₂A can work with manipulators of different degrees of freedom. We also plotted trajectory maps when domain shifts are characterised by changes in background colour. The results (reported in **Supplementary Section S5**) show that O₂A provides high rewards for desired trajectories, even when background colour changes.

Additionally, we experimented with the reach-push task using a set of manually collected video samples showing different degrees of task completion. The results reported in **Supplementary Section S6** show that O₂A reward function can successfully model the more complex task of reach-push.



5.2 Real Robot Experiment

For the real robot experiment, we use a 6DOF UR5 robotic arm attached with different end-effectors suitable for each task. All six domain shifts (see Table 4) are used for the pushing and hammering tasks. Whereas, only three domain shifts (I, V and M) are used for the sweeping and striking tasks, since others did not have meaning for

these tasks. We only used features from pool5 layer of NN:UCF101 model as the action vector, due to the high cost of running the real robot experiment. Implementation details of the STO algorithm used to generate the optimal sequence of controls are explained below.

Briefly, we begin with an initial candidate control sequence. We execute this sequence using the manipulator to generate an

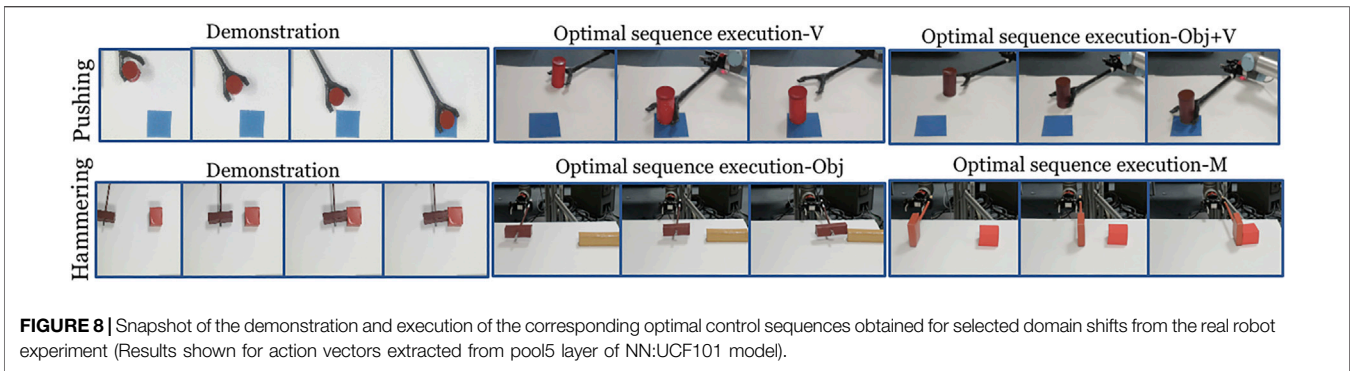


FIGURE 8 | Snapshot of the demonstration and execution of the corresponding optimal control sequences obtained for selected domain shifts from the real robot experiment (Results shown for action vectors extracted from pool5 layer of NN:UCF101 model).

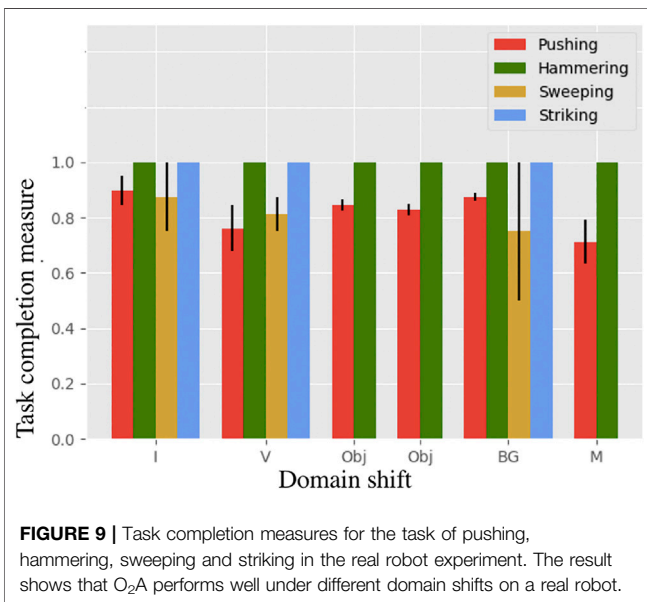


FIGURE 9 | Task completion measures for the task of pushing, hammering, sweeping and striking in the real robot experiment. The result shows that O₂A performs well under different domain shifts on a real robot.

initial cost. Thereafter, at each iteration we create eight random control sequences by adding Gaussian noise to the candidate sequence from the previous iteration and execute them using the real robot. At the end of each iteration, we pick the control sequence with the minimum cost calculated using Eq. 3. Then we set it as the new candidate sequence thereby iteratively reducing the cost. Each control sequence has a length of 16 steps and observed image frames are collected after each step. The action vector is then extracted from these frames at the end of the sequence execution to calculate the cost. The initial control sequence is initialised by providing a near solution path, following common practices in literature (Sermanet et al., 2017). The near path solution is obtained by manually setting the sequence start and end points, and interpolating between them. The Gaussian noise added has a mean zero and the standard deviation is set as hyper-parameter for each task. The detailed step by step STO algorithm is given in the **Supplementary Section S7**.

Each experiment is run two times with 10 iterations each. In **Figure 8** the snapshots of executions of optimal control sequences obtained for the selected setups are given. The average task

completion measures for the optimal control sequences are shown in **Figure 9**. Our method achieves good task completion measures for different domain shifts. This shows the effectiveness of O₂A in learning tasks on a real robot. Videos of all the results of the real robot experiments, including demonstrations are available in our project-page.

6 CONCLUSION

We have presented O₂A, a method for observational learning of robotic manipulation tasks from a single (one-shot) demonstration. The method works by extracting a perceptual representation (the action vector) from videos using a pre-trained action vector extractor. Our analysis shows that the pre-trained action vector extractor can generalise to unseen robotic manipulation tasks. Also experiments in simulation and with a real robot show that O₂A can perform well under different domain shifts and outperforms baseline approaches.

A limitation in our work is the number of trial executions required to learn a task. It would be interesting to see if we can map the action vector from the demonstration directly to an initial near optimal solution. Sim-to-real (Jeong et al., 2020) approaches could also be used to speed up the real robot experiments. Also, O₂A is currently evaluated with 2D manipulation tasks. Using 3D tasks such as stacking or grasping would help to further understand the strengths and limitations of the proposed method. Another future direction will be to use additional sensing modalities like touch or audio for situations where the demonstrations are not visually observable (e.g. due to occlusion). Also, it would be interesting to study pre-training on generic action datasets for other robotic manipulation problems. Such pre-training could potentially address the lack of large ImageNet (Simonyan and Zisserman, 2015) like datasets of robotic manipulation task videos. Finally it would be exciting to extend O₂A to multi-step manipulation tasks. One approach to tackle this could be to decompose these tasks into single-step tasks learnt using the current method, within a curriculum learning framework.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

LP conceptualized, designed and implemented the project and drafted the paper. WA assisted in the real robot experiment. DH and RF provided supervision for the project. All authors contributed to the manuscript revision, read, and approved the submitted version.

FUNDING

The work is supported by ESPRC Self Repairing Cities project (grant reference no: EP/N010523/1) and Leeds Anniversary Research Scholarship (2016) awarded to Leo Pauly.

REFERENCES

- Agboh, W. C., and Dogar, M. R. (2018). "Pushing Fast and Slow: Task-Adaptive Planning for Non-prehensile Manipulation Under Uncertainty," in Springer Proceedings in Advanced Robotics (SPAR) series on 2018 Workshop on the Algorithmic Foundations of Robotics (WAFR 2018) (Springer).
- Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics autonomous Syst.* 57, 469–483. doi:10.1016/j.robot.2008.10.024
- Athiwaratkun, B., and Kang, K. (2015). *Feature representation in convolutional neural networks*. *arXiv preprint arXiv:1507.02313*.
- Atkeson, C. G., and Schaal, S. (1997). Robot learning from demonstration. *ICML (Citeseer)* 97, 12–20.
- Azizpour, H., Razavian, A. S., Sullivan, J., Maki, A., and Carlsson, S. (2015). Factors of transferability for a generic convnet representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 38, 1790–1802. doi:10.1109/TPAMI.2015.2500224
- Bakker, P., and Kuniyoshi, Y. (1996). "Robot see, robot do: An overview of robot imitation," in *AISB96 Workshop on Learning in Robots and Animals*, 3–11.
- Bandura, A. (1986). *Social foundations of thought and action : a social cognitive theory*. New Jersey, United States: Prentice-Hall.
- Benesty, J., Chen, J., Huang, Y., and Cohen, I. (2009). *Noise reduction in speech processing*. Germany: Springer, 1–4. doi:10.1007/978-3-642-00296-0_5 Pearson correlation coefficient.
- Borsa, D., Heess, N., Piot, B., Liu, S., Hasenclever, L., Munos, R., et al. (2019). Observational learning by reinforcement learning. *Proc. Int. Jt. Conf. Autonomous Agents Multiagent Syst. AAMAS 2*, 1117–1124.
- Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., et al. (2016). *Openai gym*. *arXiv preprint arXiv:1606.01540*.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., et al. (2020). Language models are few-shot learners. *arXiv*. [Preprint arXiv:2005.14165].
- Cabrera, M. E., and Wachs, J. P. (2017). A human-centered approach to one-shot gesture learning. *Front. Robot. AI* 4, 8. doi:10.3389/frobot.2017.00008
- Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Dhariwal, P., et al. (2020). *Generative pretraining from pixels* in Proceedings of the 37th International Conference on Machine Learning.
- Dalal, N., and Triggs, B. (2005). "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition* (New Jersey, United States: IEEE Computer Society Conference on (IEEE)), 1, 886–893. 2005. CVPR 2005.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). *Bert: Pre-training of deep bidirectional transformers for language understanding*. *arXiv preprint arXiv:1810.04805*.
- Dillmann, R. (2004). Teaching and learning of robot tasks via observation of human performance. *Robotics Autonomous Syst.* 47, 109–116. doi:10.1016/j.robot.2004.03.005
- Dragan, A. D., and Srinivasa, S. S. (2012). Online customization of teleoperation interfaces. *Proc. - IEEE Int. Workshop Robot Hum. Interactive Commun.*, 919–924. doi:10.1109/ROMAN.2012.6343868

ACKNOWLEDGMENTS

The authors would like to appreciate Mohammed abdellathif, Matteo Leonetti, Mehmet Doger, Wissam Bejjani, Rebecca Stone, Mohammed Alghamdi, Hanh Tran and Mohammad Kaykanloo for fruitful discussions on our work.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: <https://www.frontiersin.org/articles/10.3389/frobot.2021.686368/full#supplementary-material>

- Duan, Y., Andrychowicz, M., Stadie, B., Ho, O. J., Schneider, J., Sutskever, I., et al. (2017). "One-shot imitation learning," in *Advances in neural information processing systems*, 1087–1098.
- Field, M., Stirling, D., Naghdy, F., and Pan, Z. (2009). "Motion capture in robotics review," *Motion capture in robotics review* in 2009 IEEE International Conference on Control and Automation (Paris, France: ICCA), 1697–1702. doi:10.1109/ICCA.2009.5410185
- Finkel, R., Taylor, R., Bolles, R., Paul, R., and Feldman, J. (1975). "An overview of al, a programming system for automation," in Proceedings of the 4th international joint conference on Artificial intelligence, 758–765.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. Massachusetts, United States: MIT press.
- Gupta, A., Eppner, C., Levine, S., and Abbeel, P. (2016). "Learning dexterous manipulation for a soft robotic hand from human demonstrations," in IEEE International Conference on Intelligent Robots and Systems 2016-Novem (New Jersey, United States: IEEE), 3786–3793. doi:10.1109/IROS.2016.7759557
- Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. (2018). "Deep reinforcement learning that matters," in *AAAI*.
- Hubert, L., and Arabie, P. (1985). Comparing partitions. *J. classification* 2, 193–218. doi:10.1007/bf01908075
- Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2002). Movement imitation with nonlinear dynamical systems in humanoid robots. *Proc. - IEEE Int. Conf. Robotics Automation 2*, 1398–1403. doi:10.1109/robot.2002.1014739
- Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2001). Trajectory formation for imitation with nonlinear dynamical systems. *IEEE Int. Conf. Intell. Robots Syst. 2*, 752–757. doi:10.1109/iro.2001.976259
- Ikeuchi, K., and Suehiro, T. (1994). Toward an assembly plan from observation. I. Robot recognition with polyhedral objects. *IEEE Trans. Robot. Automat.* 10, 368–385. doi:10.1109/70.294211
- Jeong, R., Ayta, Y., Khosid, D., Zhou, Y., Kay, J., Lampe, T., et al. (2020). "Self-supervised sim-to-real adaptation for visual robotic manipulation," in 2020 IEEE International Conference on Robotics and Automation (ICRA)(New Jersey, United States: IEEE), 2718–2724. doi:10.1109/icra40945.2020.9197326
- Jia, Z., Lin, M., Chen, Z., and Jian, S. (2020). *Vision-based robot manipulation learning via human demonstrations*. *arXiv preprint arXiv:2003.00385*.
- Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P., and Schaal (2011). Stomp: Stochastic trajectory optimization for motion planning. *2011 IEEE Int. Conf. robotics automation (Ieee)*, 4569–4574. doi:10.1109/icra.2011.5980280
- Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., et al. (2019). *Big transfer (bit): General visual representation learning*. *arXiv preprint arXiv:1912.11370*.
- Kuniyoshi, Y., Inaba, M., and Inoue, H. (1994). Learning by Watching: Extracting Reusable Task Knowledge from Visual Observation of Human Performance. *IEEE Trans. Robot. Automat.* 10, 799–822. doi:10.1109/70.338535
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature* 521, 436–444. doi:10.1038/nature14539
- Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., et al. (2015). "Continuous control with deep reinforcement learning," in International Conference on Learning Representations.

- Liu, Y., Gupta, A., Abbeel, P., and Levine, S. (2018). Imitation from Observation: Learning to Imitate Behaviors from Raw Video via Context Translation. *Proc. - IEEE Int. Conf. Robotics Automation*, 1118–1125. doi:10.1109/ICRA.2018.8462901
- Nair, V., and Hinton, G. E. (2010). “Rectified linear units improve restricted boltzmann machines,” in *ICML*.
- Pathak, D., Mahmoudieh, P., Luo, G., Agrawal, P., Chen, D., Shentu, Y., et al. (2018). “Zero-shot visual imitation,” in *International Conference on Learning Representation*. doi:10.1109/cvprw.2018.00278s
- Pauly, L. (2018). *Defining the problem of observation learning*. *arXiv preprint arXiv:1808.08288*.
- Pauly, L. (2021). *Seeing to learn: Observational learning of robotic manipulation tasks*. Technical Report, University of Leeds.
- Peng, X. B., Abbeel, P., Levine, S., and van de Panne, M. (2018). Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graphics (Tog)* 37, 143. doi:10.1145/3197517.3201311
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., and Lee, K., et al. (2018). Deep contextualized word representations. In *Proc. NAACL-HLT*. 2227–2237. doi:10.18653/v1/n18-1202
- Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training OpenAI. Technical Report.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners OpenAI. Technical Report.
- Ramirez-Amaro, K., Beetz, M., and Cheng, G. (2017). Transferring skills to humanoid robots by extracting semantic representations from observations of human activities. *Artif. Intelligence* 247, 95–118. doi:10.1016/j.artint.2015.08.009
- Sermanet, P., Lynch, C., Chebotar, Y., Hsu, J., Jang, E., Schaal, S., et al. (2018). “Time-contrastive networks: Self-supervised learning from video,” in 2018 IEEE International Conference on Robotics and Automation (New Jersey, United States: (ICRA) (IEEE)), 1134–1141. doi:10.1109/icra.2018.8462891
- Sermanet, P., Xu, K., and Levine, S. (2017). Unsupervised perceptual rewards for imitation learning. *Proc. Robotics: Sci. Syst. (Rss)*. doi:10.15607/rss.2017.xiii.050
- Sharma, P., Mohan, L., Pinto, L., and Gupta, A. (2018). *Multiple Interactions Made Easy (MIME): Large Scale Demonstrations Data for Imitation*. *Conference on robot learning*, 906–915.
- Sharma, P., Pathak, D., and Gupta, A. (2019). “Third-person visual imitation learning via decoupled hierarchical controller,” in *Advances in Neural Information Processing Systems*, 2597–2607.
- Sieb, M., Xian, Z., Huang, A., Kroemer, O., and Fragkiadaki, K. (2020). “Graph-Structured Visual Imitation,” in *Conference on Robot Learning (PMLR)*, 979–989.
- Simonyan, K., and Zisserman, A. (2015). “Very deep convolutional networks for large-scale image recognition,” in *International Conference on Learning Representations*.
- Smith, L., Dhawan, N., Zhang, M., Abbeel, P., and Levine, S. (2019). *Avid: Learning multi-stage tasks via pixel-level translation of human videos*. *arXiv preprint arXiv:1912.04443*.
- Soomro, K., Zamir, A. R., and Shah, M. (2012). *Ucf101: A dataset of 101 human actions classes from videos in the wild*. *arXiv preprint arXiv:1212.0402*.
- Stadie, B. C., Abbeel, P., and Sutskever, I. (2017). *Third person imitation learning* in 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings (OpenReview.net).
- Tipping, M. E., and Bishop, C. M. (1999). Probabilistic principal component analysis. *J. R. Stat. Soc B* 61, 611–622. doi:10.1111/1467-9868.00196
- Todorov, E., Erez, T., and Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ Int. Conf. Intell. Robots Syst.* 5026–5033. doi:10.1109/IROS.2012.6386109
- Torabi, F., Warnell, G., and Stone, P. (2019). “Recent advances in imitation learning from observation,” in *IJCAI International Joint Conference on Artificial Intelligence 2019-Augus*, 6325–6331. doi:10.24963/ijcai.2019/882
- Tow, A., Sünderhauf, N., Shirazi, S., Milford, M., and Leitner, J. (2017). “What would you do? acting by learning to predict,” in *International Conference on Intelligent Robots and Systems*.
- Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). “Learning spatiotemporal features with 3d convolutional networks,” in *Computer Vision (ICCV), 2015 IEEE International Conference on (IEEE)*, 4489–4497. doi:10.1109/iccv.2015.510
- Xie, Y., and Richmond, D. (2018). “Pre-training on grayscale imagenet improves medical image classification,” in *Proceedings of the European Conference on Computer Vision (ECCV)*. 0–0.
- Yang, S., Zhang, W., Lu, W., Wang, H., and Li, Y. (2019). Learning Actions from Human Demonstration Video for Robotic Manipulation. *IEEE Int. Conf. Intell. Robots Syst.*, 1805–1811. doi:10.1109/IROS40897.2019.8968278
- Yu, T., Abbeel, P., Levine, S., and Finn, C. (2018a). *One-shot hierarchical imitation learning of compound visuomotor tasks*. *arXiv preprint arXiv:1810.11043*.
- Yu, T., Finn, C., Dasari, S., Xie, A., Zhang, T., Abbeel, P., et al. (2018b). One-shot imitation from observing humans via domain-adaptive meta-learning. *Proc. Robotics: Sci. Syst. (Rss)*. doi:10.15607/rss.2018.xiv.002
- Zhang, H., and Nikolaidis, S. (2019). *Robot learning and execution of collaborative manipulation plans from youtube videos*. *arXiv preprint arXiv:1911.10686*.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher’s Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Copyright © 2021 Pauly, Agboh, Hogg and Fuentes. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.