What Would You Do? Acting by Learning to Predict

Adam W. Tow, Niko Sünderhauf, Sareh Shirazi, Michael Milford, Jürgen Leitner

***Abstract*— We propose to learn tasks directly from visual demonstrations by learning to predict the *outcome* of human and robot actions on an environment. We enable a robot to physically perform a human demonstrated task without knowledge of the thought processes or actions of the human, only their visually observable state transitions. We evaluate our approach on two table-top, object manipulation tasks and demonstrate generalisation to previously unseen states. Our approach reduces the priors required to implement a robot task learning system compared with the existing approaches of Learning from Demonstration, Reinforcement Learning and Inverse Reinforcement Learning.**

I. Introduction

Existing approaches to robot task learning generally fall under three distinct areas: Learning from Demonstration (LfD), Reinforcement Learning (RL) and Inverse Reinforcement Learning (IRL). Each approach comes with a key limitation: LfD approaches require a mapping between demonstrator kinematics and robot learner kinematics [1], RL approaches require access to an oracle that provides rewards to the robot learner [2], and IRL approaches require knowledge of both the states and actions executed by the demonstrator [3]–[5]. Due to these limitations, neither RL, IRL or LfD approaches are suited for learning a task from human visual demonstrations alone - see Figure 1.

This work is motivated by our hypothesis that many tasks can be learned by imitating the state transitions of a demonstrator alone. We investigate the case of a human demonstrator and robot learner, where the robot is able to observe the outcomes of the human actions. Robots that can learn from human demonstrations are unquestionably a desire of many roboticists. However, to be useful in real world settings, some specific traits of any such approach are required. Firstly, the robot should generalise human demonstration sequences to unseen states; i.e. predict the outcome of a humans actions in states not visited during the demonstrations. Secondly, human demonstrations should be robot-agnostic; i.e. no knowledge or access to the target robot is required to record task demonstrations. Thirdly, the approach should be task-agnostic; i.e. the robot can learn new tasks provided new demonstrations alone.

We present a novel approach, termed Learning from Prediction (LfP), that addresses these requirements. Specifically, we propose to learn tasks directly from visual demonstrations by learning to predict the *outcomes* of a humans actions on an environment. Operating on visual demonstrations allows for a wide range of avenues for obtaining human task demonstrations and quite naturally leads to the setting of **state** as an RGB image and **task** as sequences of RGB images.

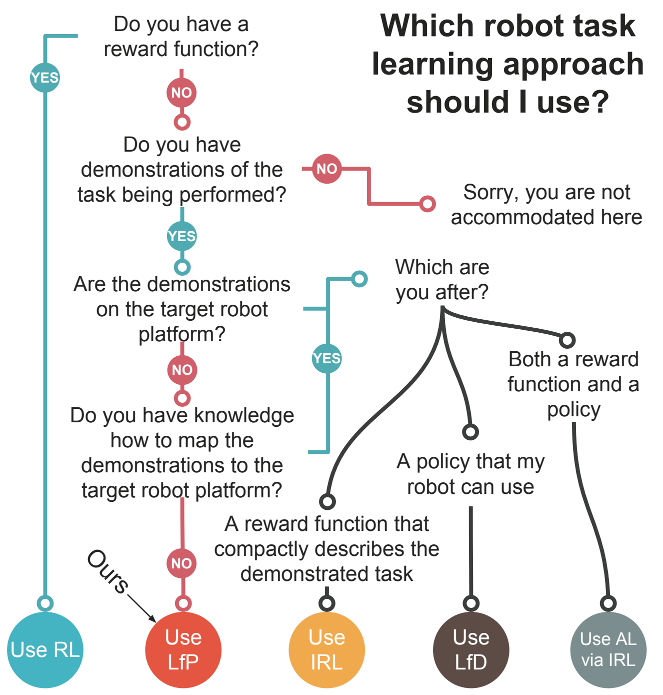

Which robot task learning approach should I use?
Do you have a reward function?
YES
NO
Do you have demonstrations of the task being performed?
NO
Sorry, you are not accommodated here
YES
Are the demonstrations on the target robot platform?
NO
YES
Which are you after?
Do you have knowledge how to map the demonstrations to the target robot platform?
NO
Ours
A reward function that compactly describes the demonstrated task
A policy that my robot can use
Both a reward function and a policy
Use RL
Use LfP
Use IRL
Use LfD
Use AL via IRL

Fig. 1. Choosing a robot task learning approach from Reinforcement Learning (RL) [2], our novel Learning from Prediction (LfP) proposed in this paper, Learning from Demonstration (LfD) [1], Inverse Reinforcement Learning (IRL) [5], [6] and Apprenticeship Learning via Inverse Reinforcement Learning (AL via IRL) [7].

Our approach equips a robot with two key capabilities that enable it to imitate a task. Firstly, provided a small number of human-performed demonstrations, the robot can predict how the environment would look *if* the human had acted in it. Secondly, provided a small number of robot-performed demonstrations, the robot can predict how the environment would look *if* it acted in it. With these capabilities, the robot at each state can exhaustively search which of its actions will bring it closest to its prediction of the humans next state.

In theory, perfect predictions will lead the robot to imitate the human perfectly across all possible states of an environment. We describe how to equip a robot with the aforementioned predictive capabilities. Our approach hinges on recent advances from the computer vision community on the problem of next-frame prediction in video sequences [8]–

The authors are with the Australian Centre for Robotic Vision (ACRV) at the Queensland University of Technology (QUT), Brisbane, Australia.
EMail: `adam.tow@qut.edu.au`

This research was supported by the Australian Research Council Centre of Excellence for Robotic Vision (ACRV) (project number CE140100016).

[15]. Currently, next-frame predictors receive a sequence of images to predict the next image. To be useful for performing a task, we require that predictions can be made from a single image i.e. to get the robot moving from the starting location. We show that the PredNet [8] next-frame predictor can be trained to operate on single images, if the sequences they are trained on are deterministic; i.e. every state has a unique next state.

We show the feasibility of our novel LfP approach on two table-top single-object manipulation tasks, designed specifically to elucidate the desirable properties of our approach. In particular, human-executed demonstrations, generality to unseen states, flexibility to different tasks, limited setup requirements and successful human-to-robot task transfer.

II. Related Work

A. Learning from Demonstration

The objective of Learning from Demonstration (LfD) is to teach a robot a task by demonstrating that task being performed. Specifically, the objective of LfD is to obtain a policy from example state-action pairs [1]. LfD works can be categorised by the demonstration approach taken, i.e. kinesthetic demonstrations, tele-operation demonstrations, or motion capture demonstrations [1], [16]–[20]. Within LfD research, solutions to the LfD problem are generally robot-specific as a result of either the demonstration approach or assumptions to map human demonstrations to the platform [1], [21].

In [21], a mapping between human-demonstrated movement and a robot was learned. Using a Kinect sensor and a pre-defined human model, the team showed a robot that could reproduce the demonstrated motion. While the correspondence between human and robot motions was learned, the approach required both the human model and robot to have the same dimensionality.

While not explicitly LfD, [22] demonstrate one example of a robotic system that learns to perform tasks from demonstrations physically performed by humans. In particular, the team show how convolutional neural network (CNN) object and human action detectors can be used to produce manipulation action plans for a robot to reproduce demonstrated cooking tasks [22]. Notably, this approach requires a common action grammar between the humans and robot.

B. Reinforcement Learning

Reinforcement Learning is a widely researched approach for solving tasks formulated as Markov Decision Processes (MDPs) [2]. In recent years Deep Reinforcement Learning has been applied to a wide range of problems, including robotics [23]–[25]. At RL's core, an agent is tasked with learning a policy by exploring its environment and maximise the reward it receives from some oracle. The two key issues of applying RL to robotics applications are the exploration time and where the reward comes from. In [24], the issue of exploration was solved by deploying 14 identical robots for 2 months to learn the task of picking up items. The issue of providing rewards was solved by choosing a known table height to close the gripper at and threshold the distance the gripper closes to produce reward [24]. These solutions are task-specific and engineered, disallowing the lay-person from teaching a robot a new task.

C. Inverse Reinforcement Learning

One might posit: if we can learn a policy provided a reward function, can we learn a reward function provided a policy? This idea is commonly termed Inverse Reinforcement Learning (IRL) or Inverse Optimal Control (IOC) [3]. Here the objective is to learn the underlying reward function that a demonstrator is optimising [6]. While it is said that a reward function is the most compact description of a task, the objective in robotics is often to find a policy such that our robot can perform the demonstrated task. Approaches to IRL that learn both a reward function and a policy are referred to as Apprenticeship Learning via Inverse Reinforcement Learning (AL via IRL) [4].

AL via IRL has seen use in robotics, notably for aerobatic helicopter flight [7]. In this case, recordings of an expert remotely piloting a helicopter were used to learn the weights for a hand-crafted 24-feature vector that defined the reward function for the task. More recently, [3] applied IOC to a number of real world robotic manipulation tasks. In this case, a neural network was used to express a reward function, removing the need for hand-engineering reward features [3]. Demonstrations of the task being performed were provided by kinesthetic teaching and as such, these demonstrations are tied to the robot platform they were performed on.

D. Video Prediction using Deep Networks

Next-frame video prediction is an unsupervised learning problem studied in the computer vision community [8]–[15]. The objective from a computer vision perspective is to leverage the wide-spread availability of video to learn feature representations that are useful for solving other tasks, i.e. object detection. The PredNet architecture used herein is one such algorithm designed for next-frame video prediction [8]. PredNet is comprised of a number of stacked modules that attempt to predict the input to that module. PredNet is shown to perform well on both synthetic and real world tasks and can support variable length inputs at test time due to internal recurrent layers.

Video prediction techniques have also been applied to help solve Reinforcement Learning problems. [26] presented a deep architecture that could learn to predict frames in Atari 2600 games. By using the actions of the player within the network, the team were able to predict the state of the game up to 100 frames into the future. The team did not use the predictions to improve game-play performance however demonstrated that the game could be played on their predictions alone [26].

More recently, a deep architecture was presented that won the Full Deathmatch track of the Visual Doom AI Competition [27]. In their work, in-game measurements such as health, ammunition and score are combined with the current image and the current goal to inform action selection. The team use action-conditioned predictions of the in-game measurements to select the action that it predicts will bring it closest to the current goal. Note it is assumed that the goal can be represented as a function of these predicted measurements.

In the case of robotics, a robotic pushing dataset was presented in [28] alongside a new approach for predicting the appearance of the environment conditioned on a robot's actions.

E. Summary

The existing areas of LfD, RL and IRL have yet to address the problem of robots learning from demonstrations where a human performs the task and no mapping between the human and the robot is made. Applications of video prediction techniques to RL and robotics have focused on improving next-frame predictions by conditioning predictions on the agent or robots actions. An existing approach that used prediction techniques for action selection relied on additional information over raw images, with training and testing performed by the same agent [27].

We propose to learn tasks directly from visual demonstrations by learning to predict the outcome of human and robot actions on an environment, without access to the thought processes or actions of the human.

III. Learning from Prediction

We herein present the Learning from Prediction approach. For generality, we define our human demonstrator as **Expert** and our robot as **Agent** in this section.

A. The Expert

Let us assume a deterministic function $\boldsymbol{\pi}^E : \mathbb{S} \rightarrow \mathbb{U}$ describes the actions $\mathbf{u}_t \in \mathbb{U}$ chosen by an expert when in state $\mathbf{s}_t \in \mathbb{S}$, so that $\mathbf{u}_t = \boldsymbol{\pi}^E(\mathbf{s}_t)$. This is typically called a *policy*. Likewise, a probabilistic model $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{u}_t)$ describes the transition from one state into the next, given action $\mathbf{u}_t$ was executed.

The thought processes leading a human expert to choose action $\mathbf{u}_t$ are unobservable to a robotic agent. In fact, we argue that even the expert's action space $\mathbb{U}$ is unknown and inaccessible to the agent. This results in both $\boldsymbol{\pi}^E$ (the expert's internal decision process) and $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{u}_t)$ (a model of how the world reacts to the expert's actions) being inaccessible.

However, the agent can observe the state of the world while the expert is acting under its policy $\boldsymbol{\pi}^E$. The occurring state transitions $\mathbf{s}_t \rightarrow \mathbf{s}_{t+1}$ are observable, assuming the robot is equipped with the appropriate sensors. These state transitions under the policy are described by the distribution $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \boldsymbol{\pi}^E)$.

While a full probabilistic model of the true distribution $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \boldsymbol{\pi}^E)$ is hard to learn, we demonstrate that approximating a deterministic predictive model $P : \mathbb{S} \rightarrow \mathbb{S}$, so that $P(\mathbf{s}_t) = \text{argmax}_{\mathbf{s}_{t+1}}\, p(\mathbf{s}_{t+1}|\mathbf{s}_t, \boldsymbol{\pi}^E)$, is computationally tractable.

B. The Agent

Our goal is to train an agent to choose actions $\mathbf{a}_t \in \mathbb{A}$ according to a parametric policy $\pi_{\boldsymbol{\theta}}(\mathbf{s}_t)$. That is, we seek the optimal model parameters $\boldsymbol{\theta}^*$ based on an optimality criterion yet to be defined.

Notice that the agent's actions $\mathbf{a}_t$ are elements of the action space $\mathbb{A}$, while the expert's actions $\mathbf{u}_t$ are elements of $\mathbb{U}$. The two spaces $\mathbb{A}$ and $\mathbb{U}$ are not identical. This makes sense since the actions that can be performed by a human will often differ greatly from the action space available to a robot[1].

Similar to above, a model $q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ describes how the state of the world changes when the agent is acting in it. As discussed before, obtaining the full probabilistic model is intractable, but we can utilise PredNet to approximate a predictive model $Q : \mathbb{S} \times \mathbb{A} \rightarrow \mathbb{S}$, so that $Q(\mathbf{s}_t, \mathbf{a}_t) = \text{argmax}_{\mathbf{s}_{t+1}}\, q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$.

C. Finding the Optimal Policy

Since we assume the expert chooses optimal actions, we would ideally like to mimic the expert's behaviour. However, since the action spaces $\mathbb{A}$ and $\mathbb{U}$ are incompatible, and the expert's actions $\mathbf{u}_t$ are unobservable as discussed above, it is impossible to learn a direct mapping from $\mathbf{u}_t$ to $\mathbf{a}_t$.

Instead, we propose the following optimal policy:

$$\boldsymbol{\pi}^*(\mathbf{s}_t) = \underset{\mathbf{a}_t^{(i)}}{\text{argmin}}\, \tilde{P}(\mathbf{s}_t) \ominus \tilde{Q}(\mathbf{s}_t, \mathbf{a}^{(i)}) \tag{1}$$

This policy executes the optimal action $\mathbf{a}_t^*$ that minimises the difference between the *predicted outcome* of the expert acting in state $\mathbf{s}_t$, and the *predicted outcome* of the agent executing $\mathbf{a}_t$ in the current state. We write $\ominus$ above to indicate a suitable difference metric on the state space $\mathbb{S}$.

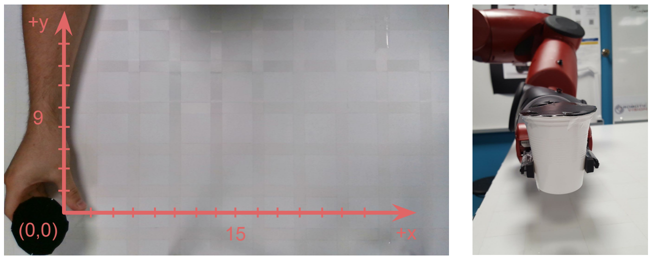
In this paper we utilise PredNet [8] to learn the approximations $\tilde{P}(\mathbf{s}_t)$ and $\tilde{Q}(\mathbf{s}_t, \mathbf{a}^{(i)})$ and train it directly on raw images. The state space $\mathbb{S}$ therefore is the space of RGB images, and we show that a suitable metric to implement the $\ominus$ operator is the mean squared error between the raw pixel values. We choose to operate on raw images to maintain the robot-agnostic and task-agnostic traits of our approach.

IV. Experimental Setup

We wish to train a robot to execute a task by imitating the state transitions of human demonstrations. The humans internal decision process and a model of how the world reacts to the humans actions is unknown. We demonstrate an approach for approximating the model $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \boldsymbol{\pi}^E)$ provided only images of human-executed task demonstrations. We also demonstrate that this approach can:

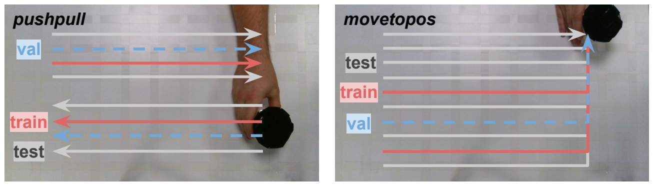
- generalise observed demonstrations to predict how the environment would change if the demonstrator had acted in an unseen state
- be applied to different tasks without modification to the approach

[1]Notice that this concept extends naturally to the case where the expert is another robot or technical system, with an action space incompatible to that of the agent that is to be trained.



(a) Discretised Task Space (b) Robot and Object

Fig. 2. (a) Human holds the object at the origin of the discretised (15x9 grid) task space. The Object can be positioned anywhere within. A tripod-mounted RGB camera provides the overhead view. (b) Learner agent is a Baxter robot, pictured holding the round, black object above the table-top environment.



(a) *pushpull* (b) *movetopos*

Fig. 3. (a) The *pushpull* task involves moving the object left or right based on its spatial location. (b) The *movetopos* task involves moving the object to a specific location; the upper right-hand corner in our case.

We restrict our investigation to two table-top, single-object manipulation tasks that demonstrate the desirable traits of our approach. The first task requires the target object to be moved to a target location and is referred to as *movetopos*. The second task requires the target object to be moved in a specific direction based on its spatial location and is referred to as *pushpull*. These tasks can be considered building blocks of more complicated tasks that involve multiple objects, such as *clean the table*.

A. The Environment

We consider the situation where a human would like to teach a robot, or robots, a set of single-object manipulation tasks. Specifically, we have a table-top environment (that is discretised into a 15x9 grid) and a round, black object that can be moved around the space (Figure 2). An overhead RGB camera is used to record task demonstrations. Demonstrations are composed of image sequences that are synchronised with state transitions through the grid. Repeatability of demonstrations and starting locations is facilitated by a white grid structure that is partially visible in the figures. We assume the object position to be discrete.

We consider only generalisation to states unseen in the demonstrations. As such, we assume that the discretised task-space, lighting conditions, object, camera and camera location remain unchanged for both human and robot interaction.

B. The Expert

We wish to approximate human task demonstrations with PredNet and demonstrate generalisation to unseen states. As mentioned earlier, we have selected two task variants based on our common grid environment to demonstrate our robot task learning approach.

The first task variant is known as *movetopos*: it demonstrates an example where an object is moved from an arbitrary start location to a desired final location. Successfully reaching the goal location from unseen start locations indicates generalisation.

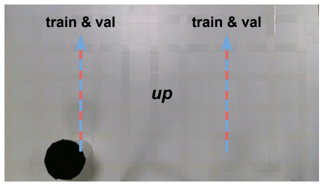
The second task variant is known as *pushpull*: it demonstrates an example where a different action must be performed based on the spatial location of the object. In our scenario, if the object is positioned in the upper half of the grid space, we wish it to move right, if in the lower half, move left. Moving in the correct direction when starting in rows of the grid unseen during the demonstrations indicates successful generalisation.

We use PredNet to approximate the predictive model $P(s_t)$ of the demonstrator. Recall that $P(s_t)$ is a prediction of the demonstrators next state provided the current state. We train PredNet to predict our human demonstrators actions by providing sequences of images that capture the desired task being performed. To collect sequences of images, we hold the object as a human and move the object through the grid space in discrete steps. An image of the environment is captured after each move is performed. Notice that the human's arm remains in the view of the scene. The specific training and validation sequences recorded for each of the two tasks can be seen in Figure 3.

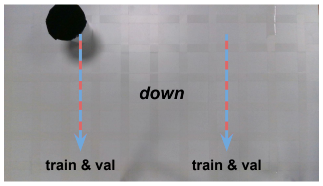
We trained one PredNet for each task and primarily maintained the hyper-parameters reported by its authors on the Kitti dataset [8]. While we use a separate PredNet for each action herein, prior approaches that produce action-conditional predictions with a single network may alternatively be used [26]–[28]. It is unclear how data-efficient action-conditional networks can be as reported applications involve training on thousands of frames. In our current setting, using multiple networks is not a limitation. We found a sequence length of five frames resulted in successfully capturing the change in direction within the *movetopos* task. We used a batch size of 4 image sequences and processed 64 image sequences per epoch (samples per epoch = 64). We trained the network for a maximum of 500 epochs, only keeping the network weights that performed best on the validation sequences. While not reported, we briefly trialled different sequence lengths and noticed the predictions no longer captured the direction change correctly, highlighting the need for a separate investigation into hyper-parameter selection and potentially into different prediction frameworks.

C. The Agent

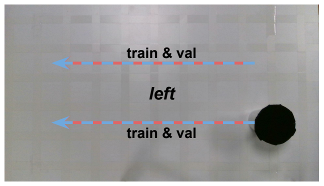
We wish to approximate the results of a robot's primitive actions on the environment with PredNet and demonstrate generalisation to unseen states. Specifically, we use a separate PredNet to approximate each of the action-specific predictive models $Q(\mathbf{s}_t, \mathbf{a}^{(i)})$ of the robot. Recall that $Q(\mathbf{s}_t, \mathbf{a}^{(i)})$ is a prediction of the robot's next state provided the current state and taking action a.



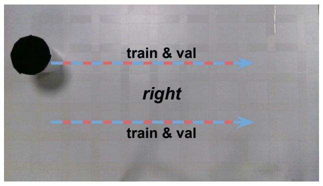
(a) Robot up primitive



(b) Robot down primitive



(c) Robot left primitive



(d) Robot right primitive

Fig. 4. Robot action primitives were trained using the demonstration trajectories depicted. The robot's arm is not present in the image to remove bias as to how the object should be held at test time. While the arm was removed artificially herein, the process can be automated with a calibrated camera, robot model and image of the empty scene.

As mentioned in Section III-B, we propose that the action spaces of the expert and agent are incompatible, $\mathbb{A} \neq \mathbb{U}$. Under this assumption, we state that predictions of the expert should be made on the current state alone, $P(s_t)$. In practice, we found PredNet was incapable of producing accurate single-image predictions for all states; leading to poor overall performance at the tasks. To remedy this result, we selected the action space $\mathbb{A} = \{up, down, left, right\}$ for the robot and aligned this with the action capabilities of the human. As such, the distance the object moves under both human and robot actions was the same. Employing this setting allowed the states physically visited by the robot to be used in the prediction of the human's next state, $P(s_{0\to t})$.

With the four primitive actions chosen, we collect training data that captures how the action primitive influences the state of the environment. The training data was collected by having the robot perform each of its action primitives twice across the grid space. The training and validation data used is depicted in Figure 4. Note for both collecting the robot primitive training data and implementing the approach on the robot, we require that the robot arm be removed from the image of the scene. By removing the robot arm, we remove any bias of the predictions prescribing the robots joint configuration while performing the task. In this work, we achieve this by taking images of the item in each grid location without the robot arm. This can trivially be replaced by a segmentation routine based on a known camera pose and robot model, coupled with a background image of the environment. We argue that this requirement is reasonable of current-day robot systems. NB: we do not require the human arm to be removed from the scene.

D. Finding an Optimal Policy

We now have four action primitive PredNets and two task-specific PredNets. Provided the current state, the action primitive PredNets provide a prediction of the environment as if the corresponding action had been performed by the robot. Ideally, the task-specific PredNets would also operate off the current state alone, and provide a prediction of the environment as if the human demonstrator had performed an action. By predicting off the current image alone, there is no requirement for the sequence of state transitions previous to the current state to align with the sequence of state transitions demonstrated by the human. As mentioned in the previous section, we boosted prediction performance by providing the robots physically executed state transitions into each subsequent prediction of the humans next state.

Algorithm 1: Learning from Prediction.

```
$state\_sequence \leftarrow []$
**for** $i \leftarrow 0$ ***to*** $sequence\_length$ **do**
    $current\_state \leftarrow capture\_image()$
    $state\_sequence.append(current\_state)$
    $P(s_t) \leftarrow predict\_expert(state\_sequence)$
    **for** $a$ ***in*** $[up, down, left, right]$ **do**
        $Q(s_t, a) \leftarrow predict\_action(current\_state, a)$
        $errors[a] \leftarrow MSE(P(s_t), Q(s_t, a))$
    **end**
    $action \leftarrow \underset{a}{\operatorname{argmin}}(errors)$
    $perform\_action(action)$
**end**
```

Our approach for applying the task-specific and action primitive predictors on a real robot platform is captured in Algorithm 1. The action primitive predictors allow the robot to choose the action which minimises the difference between its next state, and its prediction of the humans next state. Assuming successful predictions, our algorithm results with the robot successfully imitating the human on the first attempt of the task.

V. Results

We report the overall performance of Learning from Prediction on our two proposed tasks in Table I. For each task, we tested the system from every possible start location, excluding their goal locations. We define a trajectory as a sequence of steps the robot is allowed to move the object within the task environment. A trajectory is successful if the object arrives at the ground truth final location, in alignment with the demonstrations, see Figure 3.

100% of the 135 trajectories for the *movetopos* task successfully arrived at the goal location in the top right-hand corner of the task-space. 74.1% of the 112 trajectories for the *pushpull* task successfully arrived at the goal locations. In addition to this primary result, we report the performance of the approach using single-image predictions alone. We find significantly lower performance in the single-image case. Recall from Section IV-C that to improve over the single-image trajectory performance, we fed all previously visited states into each subsequent next state prediction.

Secondary to the percentage of successful trajectories, we report the percentage of trajectories that deviated from the

TABLE I

Performance at each task from all possible starting locations.

	pushpull	movetopos
Successful trajectories (predictions use previous sequence)	74.1%	100.0%
Successful trajectories (predictions from single-image)	37.5%	10.4%
Successful trajectories with no deviation from ground truth	96.4%	62.7%
Length of deviations from ground truth (additional steps taken to reach goal)	median 6, max 28, min 5	median 1, max 1, min 1

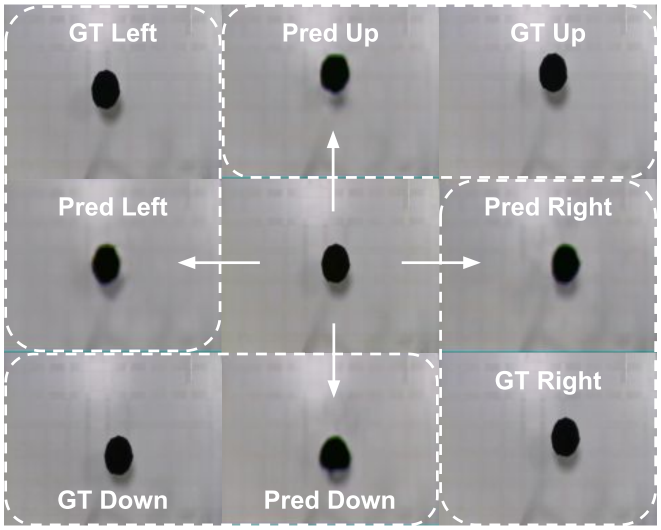
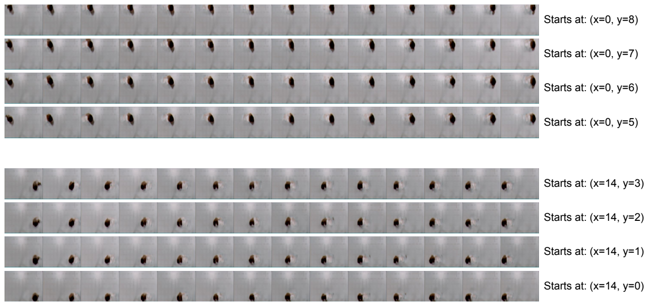
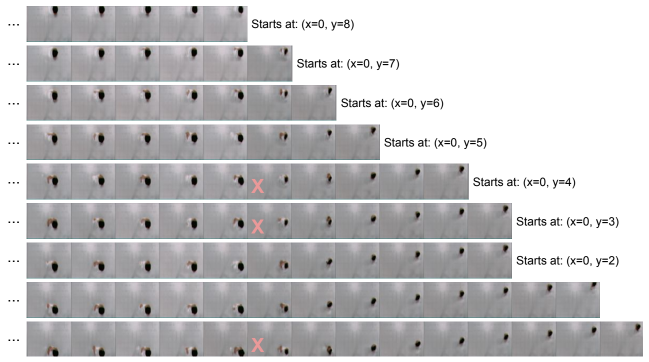


Fig. 5. Examples of action primitive predictions vs. corresponding ground truth. The four action primitive PredNets were trained as per Figure 4.



(a) *pushpull*



(b) *movetopos*

Fig. 6. Full sequences of predicted images as exemplars are shown. The red x's in the *movetopos* sequences mark predicted images that resulted in an incorrect action selection. This caused these sequences to take one additional step over the ground truth. Overall, 50 of the 134 successful *movetopos* trajectories contained an additional step at the transition point from moving rightwards to upwards. The MSE between action primitive prediction right and up at the failure locations were very close.

ground truth. Each starting location has a fixed path to the goal as demonstrated by the human. The *movetopos* task had a significant number of trajectories with an additional step. In these cases, poor predictions at the transition from moving rightwards to upwards delayed the moving up by one step. Feeding the sequence into the predictor at this failure location lead to the significant performance increase for the *movetopos* task against the single-image trajectories.

A. *Action-Primitive Prediction Performance*

Action-primitive prediction allows the robot to predict how the environment would look *if* it acted with a given action. We trained four action-primitive predictors as per Section IV-C. We show prediction performance compared against ground truth for an unseen part of the task-space in Figure 5. As can be seen, the predictions align very well with the ground truth. Only two trajectories across the task space were required to train each action primitive network - see Figure 4.

B. *Task Performance*

Task prediction allows the robot to predict how the environment would look *if* the human acted. We show a number of full sequences of predicted images as exemplars in Figure 6. As can be seen, the predictions move the block across the task-space in alignment with the human-performed demonstrations. Note that a number of the states visited in these exemplars were not visited by the human.

C. *Failed Trajectories and First-State Prediction Performance*

25.9% of the *pushpull* trajectories failed. Of these trajectories, we found that deviation from ground truth occurred at the first state with an incorrect prediction as per Table I. 26 of the trajectories failed by moving the object rightwards from the starting state (as opposed to the ground truth leftwards). 3 of the trajectories failed by moving the object upwards from the starting state. Visual observation of the single-image predictions at these starting locations highlights why the incorrect action was taken - see Figure 7.

Based on our observation that failed trajectories went wrong at the first state, we report how making a correct prediction in the first state relates to the success of that trajectory in Table II. We find that only a small number of the successful trajectories started with an incorrect initial prediction. Secondly, we find that no correct first-state prediction lead to an unsuccessful

TABLE II

WE REPORT SINGLE-IMAGE PREDICTION PERFORMANCE IRRESPECTIVE OF THE FINAL TRAJECTORIES SUCCESS. WE REPORT HOW MAKING A CORRECT PREDICTION IN THE FIRST STATE RELATES TO THE SUCCESS OF THAT TRAJECTORY.

	pushpull	movetopos
Correct single-image action predictions	71.4%	94.0%
Successful trajectories with a correct first-state prediction	96.4%	94.0%
Successful trajectories with an incorrect first-state prediction	3.6%	6.0%
Unsuccessful trajectories with a correct first-state prediction	0.0%	0.0%
Unsuccessful trajectories with an incorrect first-state prediction	100.0%	0.0%

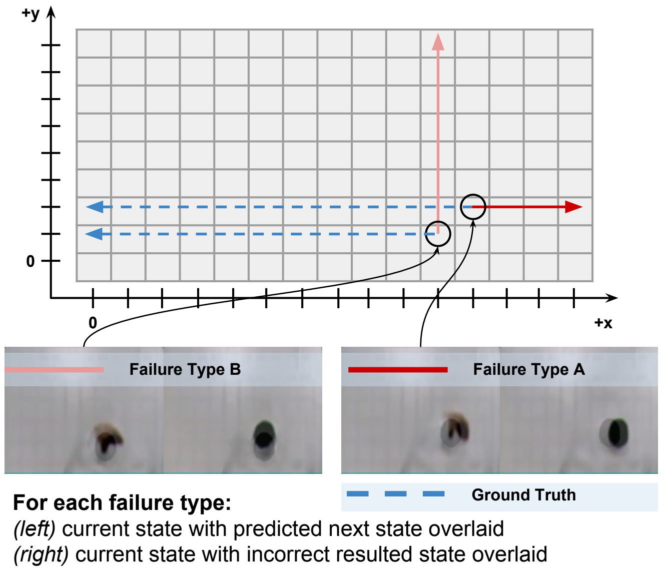

+y
0
0
+x
Failure Type B
Failure Type A
Ground Truth
For each failure type:
(left) current state with predicted next state overlaid
(right) current state with incorrect resulted state overlaid

Fig. 7. The *pushpull* task had 29 unsuccessful trajectories of a possible 112. 26 of the failures (Type A) incorrectly moved the item right on the first prediction. 3 of the failures (Type B) moved the item upwards. These two exemplars highlight the poor first-state predictions that caused these failures. Recall for the *pushpull* task the objective is to move the item left if it is located in the lower half of the task-space.

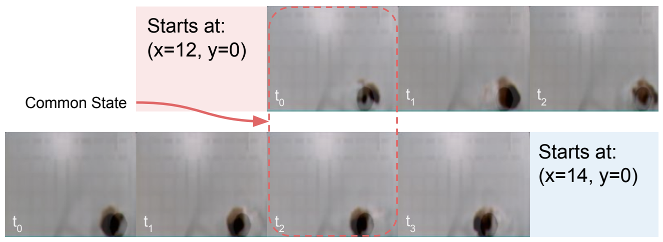


Fig. 8. Exemplar of poor first-state prediction compared against a prediction that considers the previous steps. Starting at task-space position (x=12,y=0), an incorrect initial prediction causes the object to move rightwards. A sequence starting at task-space position (x=14, y=0) has built up a correct sequence of actions by the time it arrives at (x=12, y=0) and instead makes a correct prediction and continues to move the object leftwards. Images show the current state with the predicted next state overlaid.

trajectory.

Finally, we highlight an exemplar where a first state, single-image prediction failed but a sequence passing through succeeded. As shown in Figure 8, the common state of (x=12, y=0) between two different trajectories of the *pushpull* task had two different predictions. While the sequence starting at location (x=12, y=0) failed, the sequence passing through from (x=14, y=0) succeeded in moving through.

VI. CONCLUSION AND FUTURE WORK

We presented a novel methodology for robots to learn tasks from human demonstrations called *Learning from Prediction*. The LfP approach is task-general, robot-general and human-general. These traits are desirable for two key reasons. Firstly, task demonstrations performed by a human do not require knowledge of the target robot. Robot-general demonstrations allow large, freely available video databases such as YouTube to be used. Secondly, on-robot action execution is reduced to the absolute minimum. The robot can perform the task correctly on its first attempt by predicting the outcome of all its actions before choosing to act at every state.

We used the existing PredNet architecture for predicting the outcomes of the human demonstrator and the robot's action primitives. We found PredNet could be successfully trained from only a small number of demonstrations and generalise well to unseen states. For our primary result, we used the sequence of physically visited states to improve prediction performance overall. This lead to 100% success on the *movetopos* task, accounting for all possible object starting locations.

Using the sequence of previously visited states impacted our desired trait of robot and human generality. In particular, we required that the object movement capabilities of the human and the robot be aligned. We hypothesise that predicting from a single image of the current state alone, will allow for the human and robot to have different object movement capabilities.

While the number of successful trajectories under single-image prediction was low, we found that only a small percentage of incorrect predictions occurred overall. Under single-image operation, a single incorrect prediction will cause a trajectory failure. Future work will investigate how the single image prediction performance of PredNet can be improved and potential remedies for recovering from an incorrect prediction.

Future work will also seek to apply the proposed approach to more complicated, three-dimensional tasks with varying backgrounds and distractors. While these domains are not investigated herein, we argue this work be considered a proof of concept of the approach and introduction to a new robot task learning approach we call *Learning from Prediction*.

References

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[3] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *Proceedings of the 33rd International Conference on Machine Learning*, vol. 48, 2016.

[4] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 1.

[5] E. Klein, M. Geist, B. Piot, and O. Pietquin, "Inverse reinforcement learning through structured classification," in *Advances in Neural Information Processing Systems*, 2012, pp. 1007–1015.

[6] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning." in *Icml*, 2000, pp. 663–670.

[7] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," *Advances in neural information processing systems*, vol. 19, p. 1, 2007.

[8] W. Lotter, G. Kreiman, and D. Cox, "Deep predictive coding networks for video prediction and unsupervised learning," *arXiv preprint arXiv:1605.08104*, 2016.

[9] R. Goroshin, M. F. Mathieu, and Y. LeCun, "Learning to linearize under uncertainty," in *Advances in Neural Information Processing Systems*, 2015, pp. 1234–1242.

[10] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.

[11] R. C. O'Reilly, D. Wyatte, and J. Rohrlich, "Learning through time in the thalamocortical loops," *arXiv preprint arXiv:1407.3432*, 2014.

[12] R. B. Palm, "Prediction as a candidate for learning deep hierarchical models of data," *Technical University of Denmark*, vol. 5, 2012.

[13] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory," *arXiv preprint arXiv:1511.06309*, 2015.

[14] W. R. Softky, "Unsupervised pixel-prediction," *Advances in Neural Information Processing Systems*, pp. 809–815, 1996.

[15] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using lstms." in *ICML*, 2015, pp. 843–852.

[16] S. Ekvall, "Robot task learning from human demonstration," Ph.D. dissertation, KTH, 2007.

[17] N. Koenig and M. J. Matarić, "Robot life-long task learning from human demonstrations: a bayesian approach," *Autonomous Robots*, pp. 1–16, 2010.

[18] D. Lee and C. Ott, "Incremental kinesthetic teaching of motion primitives using the motion refinement tube," *Autonomous Robots*, vol. 31, no. 2-3, pp. 115–131, 2011.

[19] M. Field, D. Stirling, Z. Pan, and F. Naghdy, "Learning trajectories for robot programing by demonstration using a coordinated mixture of factor analyzers," *IEEE transactions on cybernetics*, vol. 46, no. 3, pp. 706–717, 2016.

[20] J. Schulman, J. Ho, C. Lee, and P. Abbeel, "Learning from demonstrations through the use of non-rigid registration," in *Robotics Research*. Springer, 2016, pp. 339–354.

[21] M. Hiratsuka, N. Makondo, B. Rosman, and O. Hasegawa, "Trajectory learning from human demonstrations via manifold mapping," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 3935–3940.

[22] Y. Yang, Y. Li, C. Fermüller, and Y. Aloimonos, "Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web." in *AAAI*, 2015, pp. 3686–3693.

[23] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[24] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *arXiv preprint arXiv:1603.02199*, 2016.

[25] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017.

[26] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in atari games," in *Advances in Neural Information Processing Systems*, 2015, pp. 2863–2871.

[27] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," *arXiv preprint arXiv:1611.01779*, 2016.

[28] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Advances In Neural Information Processing Systems*, 2016, pp. 64–72.