Inverse Reinforcement Learning of Interaction Dynamics from Demonstrations

Mostafa Hussein[1], Momotaz Begum[1], Marek Petrik[2]

***Abstract*— This paper presents a framework to learn the reward function underlying high-level sequential tasks from demonstrations. The purpose of reward learning, in the context of learning from demonstration (LfD), is to generate policies that mimic the demonstrator's policies, thereby enabling imitation learning. We focus on a human-robot interaction (HRI) domain where the goal is to learn and model structured interactions between a human and a robot. Such interactions can be modeled as a partially observable Markov decision process (POMDP) where the partial observability is caused by uncertainties associated with the ways humans respond to different stimuli. The key challenge in finding a good policy in such a POMDP is determining the reward function that was observed by the demonstrator. Existing inverse reinforcement learning (IRL) methods for POMDPs are computationally very expensive and the problem is not well understood. In comparison, IRL algorithms for Markov decision process (MDP) are well defined and computationally efficient. We propose an approach of reward function learning for high-level sequential tasks from human demonstrations where the core idea is to reduce the underlying POMDP to an MDP and apply any efficient MDP-IRL algorithm. Our extensive experiments suggest that the reward function learned this way generates POMDP policies that mimic the policies of the demonstrator well.**

I. Introduction

This paper focuses on learning the dynamics of interactions between humans from observations. Such interactions take place, for example, in educational settings in which a teacher and a student goes through a sequence of steps with the goal of teaching a skill to the student [22]. Growing body of evidence gathered through numerous Wizard-of-Oz studies in the past decade shows that robots have the capacity to take the role of a teacher in structured educational intervention [27], [13], [18]. Special education teacher shortage [12] and improved learning outcomes through robot-mediated intervention (RMI) [14] motivate the need of autonomy for real-world deployment of robots in educational settings [34], [37]. Our previous works pioneered the use of learning from demonstrations (LfD) for teaching robots the steps of arbitrary educational interventions from domain experts' demonstrations [9], [10], [11]. This paper addresses a critical problem encountered by any such high-level LfD framework whose focus is to generate an abstract model of the goal-directed behaviors of a human demonstrator: *how do we define a function that will estimate the reward of being in a particular state of a sequential task in terms of highly-uncertain perceptual inputs?*

[1] Cognitive Assistive Robotics Lab, University of New Hampshire, {mhussein,mbegum}@cs.unh.edu; [2] Department of Computer Science, University of New Hampshire, mpetrik@cs.unh.edu

POMDPs are especially suitable for modeling human interactions because of the inherent uncertainty in human behavior. The goal of LfD of the task, therefore, becomes finding a policy in that POMDP which mimics the demonstrator's policy. The key requirement for policy learning is to identify a reward function. In this case, the reward function is the strategy observed by the interacting humans to take the next action as the interaction unfold with time. Even if the interaction is structured in nature, the cues and responses generated by humans are highly uncertain and handcrafting a reward function in terms of these cues is extremely difficult, if not impossible. IRL provides a natural solution to this problem where the goal is to derive the reward function underlying a set of demonstrations [41]. Unfortunately, IRL for POMDP is a less-explored domain and existing algorithms are highly computationally expensive. However, IRL algorithms for learning reward functions in MDPs are sophisticated and computationally efficient. This paper shows experimentally that learning from demonstrations in POMDP domains can be achieved effectively by reducing the POMDPs to MDPs. The learned reward functions can be used to generate POMDP policies that accurately mimic the policies of the demonstrator.

II. Background

Learning from demonstrations (LfD) or imitation learning is a popular robot learning paradigm for teaching robots new skills through human guidance [3], [7]. The key component of an LfD framework is learning the expert policy of the demonstrator. One approach for policy learning in an LfD setting is to directly mimic the exact behavior of the demonstrator without analyzing the underlying task structure and/or context. This approach is suitable for low-level LfD where the purpose typically is to learn a demonstrated motion trajectory [17]. Accordingly, reinforcement learning (RL)-based direct policy learning has been used in many low-level LfD tasks where the reward function was hand-crafted or directly extracted from the perceptual data [3], [19], [30], [20].

In the case of high-level LfD, where the goal is to generate an abstract model of the demonstrated task, direct policy learning becomes difficult due to indirect mapping between the perception and actions [7]. High-level LfD algorithms, therefore, often rely on the assumption that the best representation of an expert's behavior is the reward function, not the policy [32], [29]. Accordingly, the goal is to learn the reward function using IRL and then use it to compute the optimal the policy. IRL-based LfD has

been used in many low- and high-level LfD tasks where handcrafting of a reward function is tedious due to a high dimensional state space, e.g. continuous control of helicopter [1], parking lot navigation [2], navigating a quadruped robot across different terrains [39], human navigation behavior [28], routing preferences of drivers [40], modeling goal-directed trajectories of pedestrians [41] and user simulation in spoken dialog management systems [6].

The recent success of deep convolutional neural networks (CNNs) in approximating complex relationships among environmental features and an agent's actions [23], [33], [16] has triggered the trend of end-to-end learning of reward functions [31], [35]. In general, learning the reward function relies on the assumption that the environmental features determine the reward structure. This is even more true for CNN-based learning of reward functions which works better when the environment undergoes clearly perceivable changes caused by an agent's actions. Take, for example, the reward function for the task of *opening of a door* and *pouring to a glass* from an expert's demonstration using a CNN [31]. Here the reward starts with zero and rise proportionately as the glass gets filled up or the door opens up gradually. In contrast, the perception of human-robot interactions (such as educational intervention) is subject to uncertainties which make it difficult to establish a clear connection between observed features and an agent's actions. Since such structured interactions can be modeled as POMDPs, it opens up the possibility of using model-based IRL algorithms to extract the underlying reward function.

The primary contribution of this paper is to show, through a series of real-world experiments, how POMDPs representing human-robot structured interactions can be reduced to MDPs. Then, IRL algorithms for MDPs can be used to approximate the reward function that generates highly accurate POMDP policies. The proposed framework learns two different robot-mediated educational interventions solely from raw observations. The proposed approach is beneficial to learning many other HRI problems from demonstrations that can be represented as POMDPs.

III. Preliminaries

A. MDPs and POMDPs

An MDP is a tuple $\langle S, A, T, \gamma, R\rangle$: where S is the finite set of states; A is the finite set of actions; $T : S \times A \times S \to [0, 1]$ is the state transition function where $T(s, a, s')$ represents the probability of moving to state s' from state s by taking action a; $R : S \times A \to \mathbb{R}$ is the reward function where $R(s, a)$ represents the immediate reward generated by taking an action a in a state s; $\gamma \in [0, 1)$ is the discount factor.

The solution to an MDP is a stationary deterministic policy $\pi : S \to A$ [26]. A policy π is optimal if $\pi(s) \in \operatorname{argmax}_{a\in A} Q^\star(s, a)$ for each $s \in S$ where $Q^\star$ is defined by the following Bellman optimality condition:

$$Q^\star(s, a) = R(s, a) + \gamma \sum_{s'\in S} T(s, a, s')V^\star(s') \ .$$

The optimal value function $V^\star : S \to \mathbb{R}$ must satisfy $V^\star(s) = \max_{a\in A} Q^\star(s, a)$. We use Q^π and V^π to denote the state-action and state value functions for the policy π.

A partially observable MDP (POMDP) extends an MDP by relaxing the requirement that the present state is perfectly observable. A POMDP is defined as a tuple $\langle S, A, Z, \bar{T}, O, \gamma, R\rangle$: where $S, A, \bar{T}, R$ and γ are defined identically as in MDPs. In addition, Z is a finite set of observations and $O : S\times A\times O \to [0, 1]$ is the observation function where $O(s, a, o)$ denotes the probability of observing an observation o after executing action a and transitioning to state s.

It is well-known that any POMDP can be reduced to an equivalent belief MDP with states $\bar{S}$, actions A, transition probabilities $\bar{T}$, and rewards $\bar{R}$ [25]. Note that the actions in the belief MDP are the same as in the POMDP. The states $\bar{S} = \Delta^S$ of the belief MDP belong in the probability simplex over S. We generally use $b \in \bar{S}$ to denote a belief and $b(s)$ to represent the probability that corresponds to a state s. The transition probabilities $\bar{T}$ in the belief MDP are defined as follows:

$$\bar{T}(b, a, b') = \sum_{o\in O} \left(\mathbf{1}_{b'=b'_o} \sum_{s,s'\in S} O(s', a, o)\bar{T}(s, a, s')b(s) \right),$$

where b'_o is the belief state that follows b after an action a and an observation o and defined as:

$$b'_o(s') = \eta\, O(s', a, o) \sum_{s\in S} \bar{T}(s, a, s')b(s) \ .$$

Here, η is a normalization constant and $\mathbf{1}_{b.}$ represents and indicator function. The reward $\bar{R}(s, a)$ is defined as:

$$\bar{R}(b, a) = \sum_{s\in S} b(s)R(s, a) \ .$$

B. Inverse RL for MDPs and POMDPs

The IRL problem in the MDP context was first presented in [24] and countless methods have improved on it since; see, for example, [38], [40], [21] and references therein. The goal of IRL is to learn to emulate a policy simply by observing the execution by an expert. When the dynamics of the environment are known, it is possible to achieve impressive generalization by using the demonstration to learn the apparent rewards that drive the expert's behavior. Because we focus on relatively small problems, simple IRL methods are sufficient. We review the basic principles of IRL for MDPs and POMDPs in this section.

The necessary and sufficient condition for the reward function R of an MDP to guarantee the optimality of a policy π is [24], [32]:

$$\sum_{s\in S} \left(Q^\pi(s, a_1) - \max_{a\in A\setminus a_1} Q^\pi(s, a) \right)$$

One of the simplest IRL methods proceeds as follows [24]. Given the expert's policy $\hat{\pi}$ we seek to maximize the sum of the difference between the quality of the optimal action and the quality of the second best action the reward function

is found by solving the optimization problem using linear programming:

$$\max_{R:S\to\mathbb{R}} \quad \sum_{s=1}^{N} \min_{a\in A\setminus\{\hat{\pi}(s)\}} (P_{\hat{\pi}}(s) - P_a(s))v_{\hat{\pi}} - \lambda\|R\|_1$$
$$\text{s.t.} \quad (P_{\hat{\pi}} - P_\pi)v_{\hat{\pi}} \geq 0 \quad \forall\pi \in \Pi$$
$$|R_i| \leq R_{\max} \quad i = 1, ..., N$$

where $\hat{\pi}$ is the policy that we want to learn, P_π is the transition probability matrix for the policy π, and λ is an adjustable weight for the penalty of having too many non-zero entries in the reward function R and $v_{\hat{\pi}} = (I - \gamma P_{\hat{\pi}})^{-1}R$. Note that this simplified formulation assumes that the rewards are independent of actions and can be easily generalized to standard MDP settings [24].

The literature on IRL algorithms for POMDP is limited. We are only aware of one algorithm, which is based on finite-state controllers [8]. This algorithm converts the POMDP to a cross product MDP and then extends IRL methods for MDPs, such as [24], in this formulation. Our experimental results suggest that this approach is not viable when learning to mimic human interactions.

IV. The Problem

We focus on a real use-case of robot-mediated educational intervention where the robot will learn to take the role of a teacher to teach a child a specific skill [5]. We are particularly interested in Applied Behavior Analysis (ABA)-based intervention [15]. ABA is well known for its rigid structure and unparalleled success in teaching basic skills to children with developmental delays. In any ABA-based intervention, the interaction between two agents (a robot teacher and a student) evolve in the following way: *Command* $\rightarrow$ *Response* $\rightarrow$ *Prompt* (if required) $\rightarrow$ *Reward* $\rightarrow$ *Abort*. Here, the *Discriminative stimuli*, *Prompt*, and *Reward* are actions performed by the robot while the *Response* is executed by the child. The *Prompt* can be delivered multiple times. Such interventions are typically repeated multiple times per day for several days before a child with a developmental delay can master the target skill. This triggers the need for autonomy in the robot. For a particular intervention, Wide inter- and intra-child variations may exist in the ways the *Response* is executed, making hand-coding a tedious task. We focus on autonomous learning of two ABA-interventions that are frequently used by therapists to teach children with autism: *social greetings* and *object-naming*. In case of the *social greetings* intervention (Fig. 1(a)), the goal is to teach, following the ABA principles, how to respond to a greeting in a socially acceptable manner [4]. The goal of the *object-naming* intervention (Fig. 1(b)) is to teach to respond to query and improve the vocabulary of a child. The detailed steps of these two interventions are reported in Table I.

From a machine learning perspective, our goal is to learn the complete structure (perception-action pairs) of any ABA-based interventions entirely from observations so that the robot can deliver such interventions autonomously. Our previous work used a POMDP to model the *social greetings* intervention and performed policy selection using hand-crafted reward function and hand-picked environmental features [11]. Hand-crafted reward function resulted in erroneous policy selection. Failure in the detection of hand-picked features, a well-known problem with visual perception, further deteriorated the policy selection performance. The specific goal of this paper is to learn the reward function from demonstration data for any ABA-based intervention that can be modeled as a POMDP. We also make the perception robust through the use of a convolutional neural network (CNN).

TABLE I
Steps of Robot-mediated ABA-based Educational Interventions

Step	Controller	Social Greeting	Object Naming
1	Therapist (Robot)	Wave and say, "Hello X"	Point to the object and say, "what is this"
2	User (x)	Compliant: correct response Non compliant: No or incorrect response	Compliant: correct response Non compliant: No or incorrect response
3	Therapist (Robot)	- If compliant will say "Great job" then goto step 4 -If non compliant will give a prompt: (max 1 time) "Please say hello" then goto step 2	- If compliant will say "Great job" then goto step 4 - If non compliant will give a prompt (max 4 times): "This is (object name)" then goto step 2
4	Therapist (Robot)	End the session saying "Good bye"	End the session saying "Good bye"

V. A POMDP for ABA-based Educational Intervention

A. Model Description

We designed a POMDP to model any ABA-style educational intervention.

- States S: There are two states that a child can be in: compliant and non-compliant. The initial state $s_0 \in S$ is selected randomly.
- Actions A: There are four actions that a robot can take: *Command*, *Prompt*, *Reward*, and *Abort*. The *Reward* and *Abort* are terminal actions that ends the intervention.
- Observations Z: A predefined set of speech and visual cues that are associated with the two states of a child.
- Transition function $\bar{T}$ and observation function O: Both functions emulates the frequency and interactions occurred in the demonstration set.
- Reward function R: The goal is to learn the reward function. For comparison purpose we hand-crafted different reward functions that, in general, penalize the robot for all states-actions except for performing the correct terminal action at the appropriate state. Quickly identifying the true state is encouraged through the use of a discount factor $\gamma = 0.9$ and small penalties when performing non-terminal actions.

B. Training

1) Demonstration set: We conducted two IRB-approved user studies to create demonstration sets for training the POMDP model, one for the social greeting intervention and

the other for the object naming intervention (Fig. 3). During

(a) (b)

Fig. 1. ABA-based robot-mediated intervention scenario. The goal is to learn the entire interaction from observations for autonomous delivery by a robot

the user studies, we restricted the maximum number of prompt before executing a terminal action to one for the social greeting intervention and to five for the object naming intervention. The robot was tele-operated during these two studies.

Six college students (4 male, 2 female) without autism participated in the study. Each participant completed a minimum of 18 interactions with the teleoperated robot. The demonstration set consisted of 189 videos for the social greeting intervention and 165 videos for the object naming intervention. For the social greeting intervention, 139 videos were used for training and 50 for validation.

2) Observation processing: Observations for the object naming intervention include a verbal response of the participant whe s/he correctly/incorrectly answers the query of the robot (e.g. by saying 'a ball' or 'this is a ball' in response to the robot's question 'what is this?'). We used the on-board speech recognition module of the NAO robot to process the only observation of the object naming intervention.

Observations for the social greeting intervention include presence or absence of any combination of the following cues generated by the participant: gaze toward the robot as a contingent response to the robot's action, verbal response to the robot's greeting (e.g. 'hi', 'hello'), and hand gesture directed to the robot. We used a CNN-based framework (Fig 2) to identify the presences of any or more of these audio-visual cues. There are two separate CNNs in this framework: the first one, F_{CNN}, is trained to detect gaze and hand gesture and the second one, A_{CNN}, is trained to process verbal response. As shown in Fig. 2, both F_{CNN} and A_{CNN} have three convolution layers, one long-short term memory (LSTM) layer and one fully connected layer.

The images in the demonstration set are captured through the robot's on-board camera at 15 fps and are 640×480 in size. They are pre-processed using standard image processing techniques in such a way that the input images to the F_{CNN} only feature the face and some of the surrounding areas of the participants. Thus, the A_{CNN} is trained with gray-scale 128×8 images. The F_{CNN} is trained to infer three output classes: *Gesture detected*, *Gaze detected*, and *Nothing detected*. During autonomous execution, the inference made by the F_{CNN} for the incoming video stream is used as an observation (Z) for the POMDP model.

The audio data are captured using the robot's on-board microphone. All audio data is preprocessed using a combination of spectral subtraction and FIR filters in order to

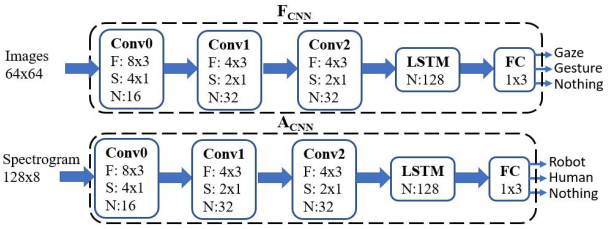


Fig. 2. The CNN-based framework for observation processing. F: filter dimension, S: Stride, N: the number of filter

TABLE II

HAND-CRAFTED REWARD FUNCTION

Action / State	Model 1	Model 2	Model 3
Command / Compliant	0	-1	-1
Command / non-compliant	0	-1	-1
Prompt / Compliant	0	0	-2
Prompt / non-compliant	0	0	-2
Reward / Compliant	1	1	10
Reward / non-compliant	0	-1	-60
Abort / Compliant	0	-1	-30
Abort / non-compliant	1	1	10
Accuracy	**75%**	**85%**	**91%**

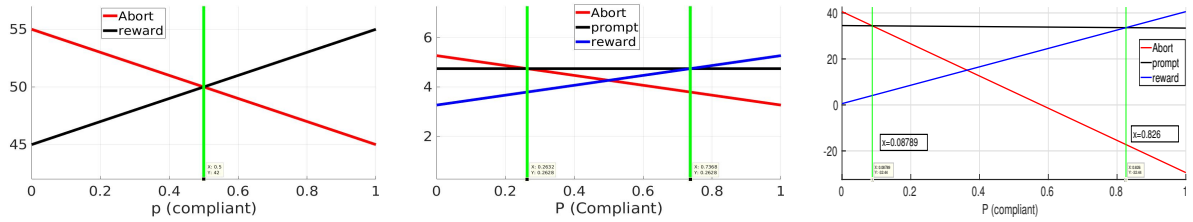
reduce the audio signal's background noise. The smoothed data is subsequently converted to a Mel-Spectrogram in order to provide a two-dimensional representation of the data [36]. Finally, the resulting Mel-Spectrograms are split into an array of frames (A) equal in length to the number of image frames that are used as the input to the F_{CNN}. Each of the frames in A has dimensions 128×8 and contains part of the previous frame in its first two columns and part of the next frame in its last three columns, to obtain a better view of the entire audio signal and include relevant patterns. The A_{CNN} is trained with the Mel-Spectrograms A and infer three output classes: *Robot speaks*, *Human speaks*, and *Nothing detected*. The *Human speaks* class corresponds to a positive response generated by the participant. During autonomous execution, the inference made by the A_{CNN} for the incoming audio stream is used as observation (Z) for the POMDP model.

We trained both networks using 140 videos from our training dataset and evaluated the model's accuracy on a set of 50 videos. The accuracy was 98.4% for the A_{CNN} and 92.6% for the F_{CNN}.

C. The Role of Reward Function:

To investigate the role of reward function on the learned policy, we hand-designed three reward models, as shown in Table II. The first reward model, Model 1, is a commonly used reward strategy in RL literature where the reward is 0 everywhere except at the terminal states where it is 1. The second model imposes some penalties for taking the incorrect action and is positive for the prompt action. The third model imposes even more penalties for taking incorrect action. We then train the POMDP with the training set using each of these three reward models.

Fig. 3 shows the $\alpha-$vectors generated from the three reward models. Any small change in reward leads to significant change in $\alpha-$vectors and hence unexpected policies.



(a) (b) (c)

Fig. 3. $\alpha-$vectors generated using different reward models (a) P(compliant state) of model 1 (b) P(compliant state) of model 2 and (c) P(compliant state) of model 3

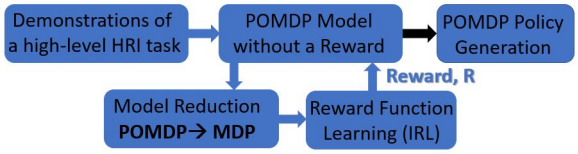


Fig. 4. The proposed model for reward learning in a POMDP

We evaluated the learned policies using the evaluation set. The accuracy is calculated as the number of times a learned policy mimics the demonstrator's policy, as observed during the demonstration. The $\alpha-$vectors corresponding to the first model represent only the reward and abort actions (fig 3(a)) while ignoring the prompt action. These conservative policy, therefore, generates very low accuracy (75%, Table II). Incurring more penalties for incorrect actions increased the accuracy to 85% (the second model) and 91% (the third model). All results were generated with ideal observation detection (i.e. all CNNs and speech recognition worked without any error).

These results shows the need of learning the reward function in order to mimic the demonstrator's policy. In many robotic application domains, including robot-mediated educational intervention, the demonstrator's policy is considered as optimal and a robot is required to exactly mimic that. Reward learning from demonstration data, therefore, is the most appropriate choice.

VI. Proposed Framework for Reward Learning in POMDP

Fig. 4 shows our proposed approach to reward learning for a POMDP representing a high-level task. Here, the core idea is to reduce the POMDP to an MDP and extract the reward function using an efficient IRL algorithm for MDPs. Through a series of experiments, we show that the reward function extracted this way, when employed in the original POMDP, generates policies that accurately mimics the demonstrators' policies. POMDP Policies generated using this proposed framework also outperforms those generated using the reward function learned through existing POMDP-IRL algorithms. We investigate two approaches for reducing a POMDP to an MDP: naive reduction and discretization. Both are discussed below.

A. Naive Reduction

The main idea of naive reduction is to map each possible observation to one MDP state, thereby eliminating the uncertainty with state estimation. Also, we assume that our observation system (described in Section V-B.2) accurately detects all observations. For example, in the case of social greeting intervention observations include gaze (G), speech (S), and hand gesture (H) generated by a child. The presence of any combination of these three cues in response to a robot action infers the child to be in a compliant state. An exception is the case where only gaze is present which is considered as non-compliant. Accordingly, the naive reduction strategy generates eight MDP states corresponding to the three cues, six of which resembles the compliant state of the original POMDP while two indicate a non-compliant state. Fig. 5 shows the POMDP model of a social greeting intervention reduced to an MDP model using the naive reduction strategy. The actions (A), and the transition function (T) for the reduced MDP are similar to the original POMDP. Once the

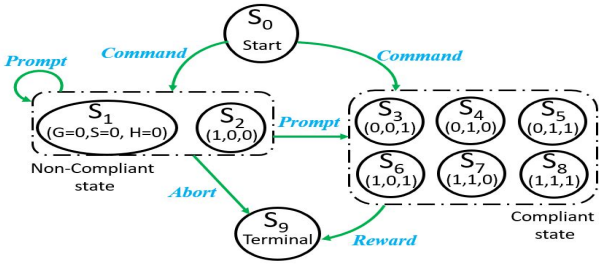


Fig. 5. Naive MDP model: 0 and 1 indicate the presence and absence of a specific observation, respectively

MDP model is generated, any existing MDP-IRL algorithm can be used to learn the reward function for this MDP. In this paper, we present experiments with the seminal IRL algorithm proposed in [24]. Finally, the reward function is employed to generate policies from the original POMDP.

B. Discretization

Discretization is a more automated process than naive reduction where we discretized the POMDP belief state to a pre-defined number n of belief segments. Each segment represents one state of an MDP. For example, for $n = 5$, we will generate an MDP with 5 states from our original 2-state POMDP where the states are: b_0, $b_{0.25}$, $b_{0.5}$, $b_{0.75}$, and b_1. Here, b_0 represents a state where a child is believed to be 0% compliant and 100% non-compliant, and so on. The states b_0 and b_1 are considered to be terminal. An initial state s_{start} is also added to start the model. The actions A are the same as those in the original POMDP. The transition function ($\hat{T} = P(b'|b, a)$) is calculated as follows to emulates the

interactions that occur between different belief states with completely unbiased state transition where the unused states are distributed with equal probability:

$$P(b''|b,a) = \sum_{o' \epsilon O} P(o'|a,b) \cdot \mathbf{1}_{b_n \in \text{argmin}_{\tilde{b}} \|b'' - b'\|_1}$$

where b' is the next belief state and b'' is the next *discretized* belief state. The probability $P(b''|b,a)$ can be computed readily using Bayes formula, see e.g. [25], as follows:

$$b(s') = P(s'|a,o,b) \cdot P(s'|a,b)$$
$$P(s'|a,b) = \sum_{s \epsilon S} P(s'|a,s) \cdot b(s)$$
$$P(o'|a,b) = \sum_{s' \epsilon S} P(o'|s) \cdot P(s'|a,b)$$

Once we have the MDP model, any MDP-IRL algorithm can be applied to learn the reward function. We used the algorithm proposed in [24]. Finally, the generated reward is mapped to the original POMDP model as follow.

$$R(s,a) = \sum_{s \epsilon S} r(a,b) \cdot b(s) \ ,$$

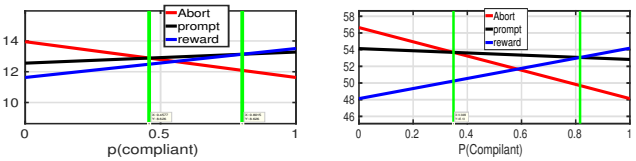
where $R(s,b)$ represents the reward function from the belief MDP model and $r(a,b)$ represents the reward function for the POMDP model.

VII. Experiments and Results

We validated the proposed approach through experiments conducted for the two HRI problems discussed in Section IV. The demonstrated data was used to design and train both the POMDP and the reduced MDP. The reward function learned from the MDP was used with the POMDP to generate policies. To assess the accuracy of these policies we organized a user study where 4 participants, who were not a part of the demonstration set, were invited to interact with the robot in the context of *social greeting* and *object naming* interventions. The robot generated policies online while analyzing the observation and interacted with the participants in a completely autonomous manner. The accuracy is defined as the number of times (in percentage) the robot delivered the correct action.

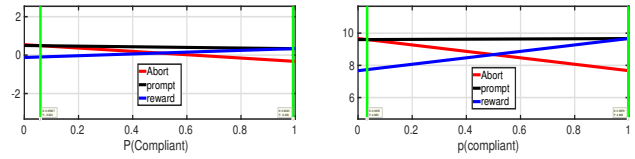
We also compared the POMDP policies generated using the proposed approach with those generated using the Two existing POMDP-IRL algorithms: dynamic programming update based approach (DP), witness update based approach (Witness) [8] and the handcrafted reward discussed earlier in section V-C (Original).

The α-vectors for the POMDPs generated using the reward functions learned from the MDP following the two methods (naive reduction and discretization) are shown in Fig. 6. As a comparison, Fig. 7 shows the α-vectors generated using the two POMDP-IRL algorithms proposed in [8]. The common problem with these vectors is that they allow the *reward* action only when the belief is extremely high about compliance (nearly 100%). Accordingly, the robot executes the *prompt* action even if a participant responded in a compliant manner.



(a) (b)

Fig. 6. α-vectors for the POMDPs generated using the proposed approach (a) MDP generated through naive reduction (b) MDP generated through discretization



(a) (b)

Fig. 7. α-vectors for the POMDP generated using (a) DP based IRL approach for POMDP (b) Witness based IRL approach for POMDP

Accuracies of the policies generated using the α-vectors in Figs. 6 and 7 are listed in Table III for the Social greeting intervention and in Table IV for the object naming intervention.

TABLE III

Social Greeting: Accuracy of different reward functions

Observation	DP	Witness	Original	Discretized	Simplified
1	87.5%	87.5%	87.5%	87.5%	100%
2	87.5%	91.6%	95.8%	100%	100%
Accuracy	**87.5%**	**89.6%**	**91.6%**	**93.75%**	**100%**

TABLE IV

Object Naming: Accuracy of different reward functions

Observation	DP	Witness	Original	Discretized	Simplified
1	50%	50%	50%	50%	100%
2	100%	75%	100%	100%	100%
3	100%	75%	100%	100%	100%
4	100%	75%	100%	100%	75%
5	50%	50%	75%	75%	75%
Accuracy	**80%**	**65%**	**85%**	**85%**	**90%**

VIII. Conclusion

In this paper, we presented a framework to learn the reward function of a POMDP representing high-level sequential tasks from demonstrations. The core idea of the proposed framework is to reduce the POMDP underlying the seqential task to a MDP and extract the rewrad function using computationally efficient MDP-IRL algorithms. Through a series of experiments with two real-world HRI tasks, we show that the POMDP policies generated using such reward functions accurately mimic a demonstrator's policies. We also demonstrate through experiments that the POMDP policies generated using our proposed framework outperforms the policies generated using existing POMDP-IRL algorithsm. The proposed framework, therfore, offers a simple yet elegent way to use POMDP models to learn high-level sequential tasks from demonstations.

Acknowledgment

This work was supported in part by the National Science Foundation (IIS-1815275).

REFERENCES

[1] Pieter Abbeel, Adam Coates, and Andrew Y Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[2] Pieter Abbeel, Dmitri Dolgov, Andrew Y Ng, and Sebastian Thrun. Apprenticeship learning for motion planning with application to parking lot navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1083–1090. IEEE, 2008.

[3] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[4] Momotaz Begum, Richard W Serna, David Kontak, Jordan Allspaw, James Kuczynski, Holly A Yanco, and Jacob Suarez. Measuring the efficacy of robots in autism therapy: How informative are standard HRI metrics'. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, pages 335–342. ACM, 2015.

[5] Momotaz Begum, Richard W Serna, and Holly A Yanco. Are robots ready to deliver autism interventions? a comprehensive review. *International Journal of Social Robotics*, 8(2):157–181, 2016.

[6] Senthilkumar Chandramohan, Matthieu Geist, Fabrice Lefevre, and Olivier Pietquin. User simulation in dialogue systems using inverse reinforcement learning. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[7] Sonia Chernova and Andrea L Thomaz. Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(3):1–121, 2014.

[8] Jaedeug Choi and Kee-Eung Kim. Inverse reinforcement learning in partially observable environments. *Journal of Machine Learning Research*, 12(Mar):691–730, 2011.

[9] M. Clark-Turner and M. Begum. Deep reinforcement learning of abstract reasoning from demonstration. In *HRI '18: ACM/IEEE International Conference on Human-Robot Interaction*, March 2018.

[10] Madison Clark-Turner and Momotaz Begum. Deep recurrent Q-Learning of behavioral intervention delivery by a robot from demonstration data. In *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 1024–1029. IEEE, 2017.

[11] Madison Clark-Turner and Momotaz Begum. Learning to deliver robot-mediated behavioral intervention. 2017.

[12] Freddie Cross. Teacher shortage areas nationwide listing 1990–1991 through 2017–2018. *U.S. Department of Education, Office of Postsecondary Education*, 2017.

[13] Lorenzo Desideri, Marco Negrini, Massimiliano Malavasi, Daniela Tanzini, Aziz Rouame, Maria Cristina Cutrone, Paola Bonifacci, and Evert-Jan Hoogerwerf. Using a humanoid robot as a complement to interventions for children with autism spectrum disorder: a pilot study. *Advances in Neurodevelopmental Disorders*, pages 1–13, 2018.

[14] J.J. Diehl, L. Schmitt, C. R. Crowell, and M. Villano. The clinical use of robots for children with autism spectrum disorders: A critical review. *Research in Autism Spectrum Disorders*, 6(1):249–262, 2012.

[15] Richard M Foxx. Applied behavior analysis treatment of autism: The state of the art. *Child and adolescent psychiatric clinics of North America*, 17(4):821–834, 2008.

[16] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Andrew Sendonaris, Gabriel Dulac-Arnold, Ian Osband, John Agapiou, et al. Deep q-learning from demonstrations. *Association for the Advancement of Artificial Intelligence (AAAI)*, 2018.

[17] Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation*, 25(2):328–373, 2013.

[18] Elizabeth S. Kim, Lauren D. Berkovits, Emily P. Bernier, Dan Leyzberg, Frederick Shic, Rhea Paul, and Brian Scassellati. Social robots as embedded reinforcers of social behavior in children with autism. *Autism and Developmental Disorders*, 43:1038 – 1049, 2013.

[19] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[20] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Robot motor skill coordination with EM-based reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3232–3237. IEEE, 2010.

[21] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 19–27, 2011.

[22] Scott R McConnell. Interventions to facilitate social interaction for young children with autism: Review of available research and recommendations for educational intervention and future research. *Journal of autism and developmental disorders*, 32(5):351–372, 2002.

[23] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[24] Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 663–670, 2000.

[25] Joelle Pineau, Geoff Gordon, Sebastian Thrun, et al. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1025–1032, 2003.

[26] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[27] Kathleen Richardson, Mark Coeckelbergh, Kutoma Wakunuma, Erik Billing, Tom Ziemke, Pablo Gomez, Bram Vanderborght, and Tony Belpaeme. Robot enhanced therapy for children with autism (dream): A social model of autism. *IEEE TEchnology and SocIETy MagazInE*, 37(1):30–39, 2018.

[28] Constantin A Rothkopf and Dana H Ballard. Modular inverse reinforcement learning for visuomotor behavior. *Biological cybernetics*, 107(4):477–490, 2013.

[29] Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the Conference on Computational Learning Theory (COLT)*, pages 101–103. ACM, 1998.

[30] Stefan Schaal, Jan Peters, Jun Nakanishi, and Auke Ijspeert. Learning movement primitives. In *Robotics research. the eleventh international symposium*, pages 561–572. Springer, 2005.

[31] Pierre Sermanet, Kelvin Xu, and Sergey Levine. Unsupervised perceptual rewards for imitation learning. *arXiv preprint arXiv:1612.06699*, 2016.

[32] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press, 1998.

[33] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 2094–2100, 2016.

[34] Joshua Wainer, Ben Robins, Farshid Amirabdollahian, and Kerstin Dautenhahn. Using the humanoid robot kaspar to autonomously play triadic games and facilitate collaborative play among children with autism. *IEEE Transactions on Autonomous Mental Development*, 6(3):183–199, 2014.

[35] Markus Wulfmeier, Dushyant Rao, Dominic Zeng Wang, Peter Ondruska, and Ingmar Posner. Large-scale cost function learning for path planning using deep inverse reinforcement learning. *The International Journal of Robotics Research*, 36(10):1073–1087, 2017.

[36] Li-Chia Yang, Szu-Yu Chou, Jen-Yu Liu, Yi-Hsuan Yang, and Yi-An Chen. Revisiting the problem of audio-based hit song prediction using convolutional neural networks. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 621–625. IEEE, 2017.

[37] Zhi Zheng, Shuvajit Das, Eric M Young, Amy Swanson, Zachary Warren, and Nilanjan Sarkar. Autonomous robot-mediated imitation learning for children with autism. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2707–2712. IEEE, 2014.

[38] Shao Zhifei and Er Meng Joo. A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 5(3):293–311, 2012.

[39] J Zico Kolter and Andrew Y Ng. The stanford littledog: A learning and rapid replanning approach to quadruped locomotion. *The International Journal of Robotics Research*, 30(2):150–174, 2011.

[40] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Association for the Advancement of Artificial Intelligence (AAAI)*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[41] Brian D Ziebart, Nathan Ratliff, Garratt Gallagher, Christoph Mertz, Kevin Peterson, J Andrew Bagnell, Martial Hebert, Anind K Dey, and Siddhartha Srinivasa. Planning-based prediction for pedestrians. In

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3931–3936, 2009.