Training a robot with evaluative feedback and unlabeled guidance signals

Anis Najar, Olivier Sigaud, Mohamed Chetouani

Training a robot with evaluative feedback and unlabeled guidance signals

Anis Najar[1], Olivier Sigaud[1] and Mohamed Chetouani[1]

Abstract— **In this paper, we present a new method for training a robot by natural interaction using evaluative feedback and unlabeled guidance signals. Feedback signals are directly mapped to reward values and used for learning both the task and the meaning of the guidance signals. The learned guidance signals are used in return to bootstrap task learning. We propose to use unlabeled guidance signals as an alternative solution to preprogrammed guidance. We evaluate our method both in simulation and on a real robot.**

I. INTRODUCTION

As more and more robots are intended to evolve in human environments either for domestic or industrial purposes, many efforts are made in designing new methods that would enable persons with no programming skills to teach them new tasks [1], [9], [16]. The main challenge is to design teaching methods that satisfy several criteria such as simplicity, efficiency and safety. In addition to these requirements, robots endowed with such learning capacities should be able to adapt to different user profiles having different preferences and teaching strategies.

The Reinforcement Learning (RL) framework provides a set of methods allowing an agent to autonomously learn a task [17]. This is done through a predefined reward function that allows the robot to optimize its behaviour. While the main advantage of RL is the autonomy of the learning process, when applied to real-world problems it presents some limitations such as the difficulty to implement a reward function [14], slow convergence and unsafe exploration [10].

The Interactive Reinforcement Learning literature deals with these problems by allowing a human to guide the learning process of the robot by providing it with teaching signals such as instructions [5], advice [4], demonstrations [1], guidance [16], [18] and evaluative feedback [9], [15].

Evaluative feedback represents an intuitive way for training a robot by evaluating its behaviour through positive and negative rewards. This way of providing rewards has some advantages over traditional RL reward functions such as being more directly informative about the optimal behaviour and easier to implement [9]. Some examples of successful implementations can be found in [9], [11], [15]. However, feedback is still limited as it is only reactive to the robot's actions, so it does not allow to anticipate them: As the robot may perform many undesired and/or unsafe actions before trying the appropriate one, feedback does not allow to limit these unsuccessful trials nor to prevent the robot from repeating them in every different state, which may result in problems in terms of efficiency and safety considerations [3].

To deal with this problem, guidance can be used in addition to evaluative feedback to constraint the exploration towards a limited set of actions [16], [18]. Classically, the meaning of guidance signals is supposed to be known by the robot before learning the task. However, as guidance signals could be task specific, it would be difficult to non expert users to program their meaning for every task. In addition, handcoded guidance would limit the possibility for different teachers to use their own preferred guidance signals.

One possible solution to overcome this limitation would be to use unlabeled guidance signals. Very few works have considered the question of learning from unlabeled teaching signals. This question is difficult since it requires to solve two problems simultaneously: learning the meaning of teaching signals and using these signals to learn the task. In [12], the robot learns to associate human gestures to actions in the context of a navigation task, but does not use them to learn the task. In [5], the authors address both questions simultaneously by using unlabeled teaching signals to learn a pick and place robotic task. In a similar approach [13], we proposed a model that learns the meaning of teaching signals from task rewards while using them to accelerate task learning.

In this paper, we propose a modified version of the model in [13] that learns the meaning of guidance signals by using evaluative feedback instead of task rewards. In this model, unlabeled guidance signals indicate the optimal action while evaluative feedback is converted into reward values. This model also extends previous research [16], [18] by relaxing the constraint of programming the meaning of guidance signals. We evaluate our model on the task proposed in [16] and we compare its performance with respect to using only evaluative feedback.

In the next section, we provide a description of our model. Then we present our scenario in Section III. Sections IV and V report respectively the results obtained in simulation and on a Baxter Research Robot. Finally, we discuss the advantages and the limitations of our model in Section VI.

II. METHOD

The main idea of our model is to learn the meaning of guidance signals while using them for guiding the learning process. On the one hand, learning the meaning of guidance signals consists in learning the correspondence between each signal and the set of actions to which it refers. In this paper, we consider the special case where each guidance signal

*This work was supported by ROMEO2 project

[1]Sorbonne Universités, UPMC Univ Paris 06, UMR 7222, ISIR, F-75005, Paris, France. anis.najar@isir.upmc.fr

corresponds to a single action. So, here guidance can be considered as instructions or demonstrations that indicate the optimal action [16]. In this case, interpreting guidance consists in finding the optimal action for each given signal. This problem is similar to finding the optimal action for each task state, but performed on an alternative state space of reduced complexity. On the other hand, using guidance for learning consists in transferring the information about the optimal action back to the original state space. In our model, these two processes are performed simultaneously (Fig. 1).

The motivation behind this idea is that in several applications, the state space is much more complex than the action space. While task states can be defined over complex and noisy sensory inputs, a limited number of actions can be sufficient to explore the entire state space. This is the case for example in industrial tasks where the robot has to perform a sequence of actions on different objects in different situations (cf. Section III). The set of actions can be the same for all objects, only their sequence may differ. So several states would share the same optimal action.

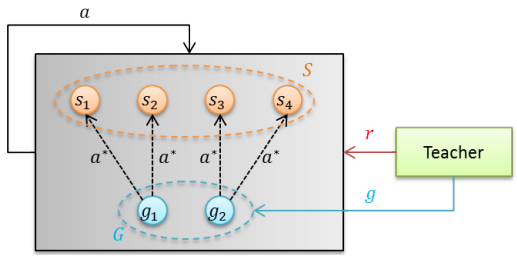


Fig. 1: An example of task with 4 states and 2 actions: A teacher can provide a guidance signal $g_i \in G$ for each state $s_i \in S$. When the robot performs action a, the teacher provides it with a reward r. This reward is used for learning the optimal action a^* associated to each guidance signal $g_i \in G$. This knowledge is then transferred to each task state $s_i \in S$.

A. Model

We model our problem within the Reinforcement Learning framework as a *Markov Decision Process* (MDP) [17] represented by a tuple $< S, A, T, R, \gamma >$. S is the state space of the task. A is a predefined set of actions that the robot can perform. $T : S \times A \rightarrow S$ is a transition function between task states. $R : S \times A \rightarrow \mathbb{R}$ is a reward function and $\gamma \in [0, 1]$ is a discount factor that determines how much future decisions should be taken into account. Within this formalism, a *q-learning* agent stores a state-action value function $Q : S \times A \rightarrow \mathbb{R}$ computed by

$$Q[s, a] \leftarrow Q[s, a] + \alpha * (r + \gamma * max_{a'} Q[s', a'] - Q[s, a])$$

where $r \in R(s, a)$, $s' \in T(s, a)$ and α a learning rate.

In our model that we call SVFB (for Social Value and Feedback, Fig. 2), we use two separate MDPs: a Task Model and a Teaching Model[1]. The main difference between the two MDPs is in the state space domains. The state space S of the Task Model is defined over task states while the state space G of the Teaching Model is defined over the unlabeled guidance signals provided by the teacher. Both MDPs share the same action set A .

We use $\gamma = 0$ as discount factor for both MDPs. Many recent works point that this is more suitable for learning from evaluative feedback as it better reflects teachers expectations about how reward is processed by learning agents [6], [8], [19]. Consequently, the learning problem becomes a sequence of single-step problems and the update formula of *q-learning* can be rephrased in the form

$$Q[s, a] \leftarrow Q[s, a] + \alpha * (r - Q[s, a]).$$

The reward functions of the two MDPs are defined by the probability of receiving a positive or a negative feedback for each state-action and guidance-action pair. In a given state, if the robot performs the optimal action corresponding to the guidance signal, the teacher provides it with a positive reward $r = 1$, and a negative reward $r = -1$ otherwise.

Finally, both MDPs communicate with each other through a third element, the Contingency Model that associates each task state to the corresponding guidance signal using:

$$\hat{g}(s) = \arg\max_{g \in G} P(g|s); s \in S.$$

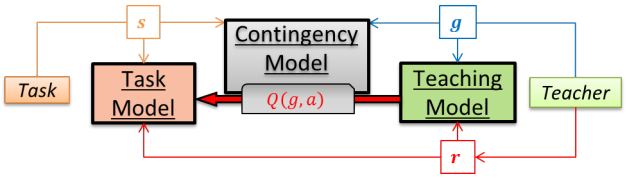


Fig. 2: The three elements of the SVFB model. The Contingency Model stores the contingency between task states s and guidance signals g. The Teaching Model learns the meaning of guidance signals using evaluative feedback r. The Task Model learns the task using both evaluative feedback r and the values learned by the Teaching Model.

B. Algorithm

Algorithm 1 shows the different steps of our method. First, the robot evaluates the task state (line 1). If it receives a guidance signal from the human, it uses it with the task state to update the Contingency Model (lines 2 to 4). Then, the guidance signal is retrieved from the Contingency Model as the most likely guidance signal for the task state (line 5). This step allows to minimize the number of interactions for the teacher by recalling previously given guidance signals. Second, if a guidance signal exists, the Task Model is updated towards the Q-values of the Teaching Model (lines 6 to 8). Then the robot performs an action (line 9). If a reward is given by the human (line 10), it is used for updating the Task Model (line 12). Finally, if a guidance signal exists, the reward is also used to update the Teaching Model (line 14).

[1]We called it Social Model in [13], but Teaching Model is more appropriate.

Algorithm 1 SVFB

$s \leftarrow$ task state
$g \leftarrow$ guidance signal
if $g \neq null$ **then**
    update_contingency(s,g)
$g \leftarrow$ get_contingency(s)
if $g \neq null$ **then**
    for all actions a in A **do**
        $Q^S[s,a] \leftarrow Q^S[s,a] + \alpha * (Q^G[g,a] - Q^S[s,a])$
$a \leftarrow \arg\max_{a \in A} Q^S[s,a]$
$r =$ teacher reward
if r $\neq$ null **then**
    $Q^S[s,a] \leftarrow Q^S[s,a] + \alpha * (r - Q^S[s,a])$
    if $g \neq null$ **then**
        $Q^G[g,a] \leftarrow Q^G[g,a] + \alpha * (r - Q^G[g,a])$

III. Scenario

To evaluate our model, we consider the object sorting domain described in [16] in which the robot has to learn to sort different objects according to their type.

A. Experimental setup

The experimental set-up (Fig. 3) is composed of a Baxter Research Robot facing a table on top of which we place three magnets. The magnets allow to place objects at three different positions on the table: left, middle and right. A webcam is placed between the robot and the table allowing to take pictures of the objects placed at the middle position. A Microsoft Kinect[2] V2 sensor is placed on top of the robot's head and used for extracting guidance and feedback signals from the teacher. The robot is controlled using the ROS architecture presented in [13]. The screen on the robot's head is used as a transparency device for displaying some relevant information such as the task state, the associated guidance signal, the performed action and the perceived reward.

B. Task domain

The workspace is divided into three zones: z_1, z_2 and z_3. When the teacher places an object in z_2, the robot must pick it up and then place it in the appropriate zone. Unicolor objects ($Plain$) must be placed in z_1 while objects with patterns ($Pattern$) must be placed in z_3 (Fig. 4).

The task state space is defined by the tuple $S = (L_{lh}, L_{rh}, L_o, D)$. $L_{lh} = \{z_2, z_3\}$ and $L_{rh} = \{z_1, z_2\}$ represent the locations of the robot's hands. Each hand can be located in two different positions above z_1, z_2 and z_3. $L_o = \{z_1, z_2, z_3, lh, rh, Unknown\}$ describes the position of the object which can be located in one of the three zones or in one of the robot's hands. The object position may also be unknown to the robot. Finally, D describes the features of the object[3].

[2]https://dev.windows.com/en-us/kinect, accessed 20-12-2014

[3]We rely on a more compact representation than [16] that removes redundancies between some state descriptors.

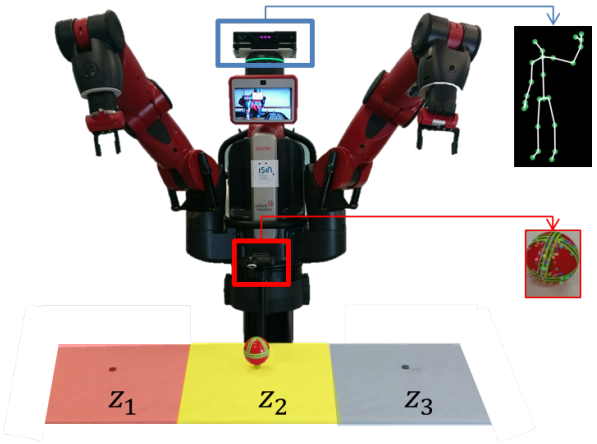


Fig. 3: Experimental setup. The table is divided into three regions: z_1, z_2 and z_3. A webcam allows to extract the features of an object placed in z_2. A Kinect V2 sensor extracts feedback and guidance signals from the teacher.

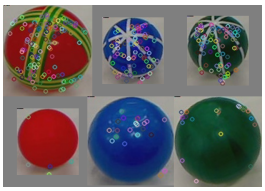


Fig. 4: Objects used in the experiments. We consider two types of objects: $Plain$ (bottom) and $Pattern$ (top). In addition, we consider two different sizes and three colors.

For our experiments, we follow the same protocol as [16] by considering two different conditions:

Small state space: In this condition, the object descriptor $D \in \{Plain, Pattern, Unknown\}$ contains a single variable based on the number of Speeded-Up Robust Features (SURF) [2] descriptors of the object. This variable can have three possible values: $Plain$, $Pattern$ and $Unknown$. Theses values are obtained by thresholding the number of extracted SURF descriptors. If this number is less than 50, the object is considered as $Plain$. Otherwise it is considered as $Pattern$. The number of different task states resulting from this representation is 72.

Large state space: In this condition, the object descriptor $D = (SURF, COLOR, SIZE)$ contains three variables: $SURF \in \{Plain, Pattern, Unknown\}$ describes the number of SURF descriptors as in the previous condition, $COLOR \in \{Red, Green, Blue, Unknown\}$ describes the dominant color of the object that can be red, green or blue, and $SIZE \in \{Large, Small, Unknown\}$ describes the area of the bounding box of the object. This representation yields a total number of task states of 864.

Finally, the robot is able to perform nine elementary actions that are necessary for completing the task (11 in [16]) $A = \{TakePicture, xMoveLeft, xMoveRight, xPick, xPlace\}$, where $x \in \{LeftHand, RightHand\}$.

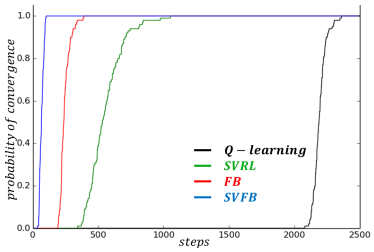
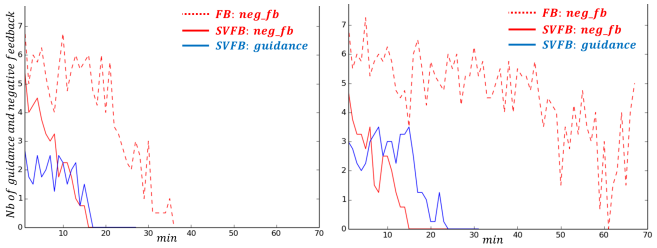


Fig. 5: Probability to converge before n steps. For each experiment, we cumulate the number of steps over the trials until convergence. For each model, we compute the density histogram of this measure over all experiments then convert it into a cumulative distribution function.



(a) Small state space (b) Large state space

Fig. 6: Number of guidance (blue) and negative feedback (red) over time.

C. Teaching protocol

For training the robot, the teacher can use either only feedback or feedback plus guidance. These information are extracted from the Kinect sensor. Feedback is divided in two categories $fb \in \{head\_nod, head\_shake\}$. By convention, head nods are converted into positive reward while head shakes are converted into negative reward. Guidance signals are defined over the teacher hand gestures. For each hand, the system recognizes five different states $h \in \{pointing\_right, pointing\_left, pointing\_middle, raised\_open, raised\_closed\}$ which allows 35 possible guidance signals using either one or both hands. In this paper, we use one signal per action, so we only use nine guidance signals.

IV. Results in simulation

In order to evaluate our model (SVFB), we compare its performance in simulation with respect to the model using only feedback (FB) as in [16]. In addition, we report the performance of Q-learning and the previous version of our model (referred to as SVRL). Both Q-learning and SVRL rely on a goal-based reward function that is defined as follows: the robot receives a positive reward $r = 1$ for placing the object in the correct zone, a negative reward $r = -1$ for placing it in the wrong zone and a reward $r = 0$ for all other transitions.

In order to compare FB and SVFB in the same conditions, we assume that the teacher provides feedback for every action. However, guidance in SVFB is provided only once for each different state. In the simulated task, we consider only the large state space condition without the $TakePicture$ action, which means that the descriptors of the object are always known to the robot. For FB and SVFB, we use $\gamma = 0$, $\alpha = 0.3$ and a greedy action selection strategy as exploration can be controlled by human rewards. For Q-learning and SVRL we use $\alpha = 0.3$, $\gamma = 0.75$ and an $\epsilon-$greedy action selection strategy with $\epsilon = 0.1$ and a decay parameter for ϵ after each step of $\delta_\epsilon = 0.001$ (ϵ reaches 0 after 100 steps). All Q-values are initialized to 0.5. The results for each model are averaged over 100 runs with 200 trials each.

Figure 5 reports the probability for each model to converge before n steps. The results show that Q-learning converges in at most 2359 steps, SVRL converges in at most 1052 steps, FB converges in at most 391 steps and finally our model converges in at most 102 steps.

In order to evaluate the cost of our method with respect to the feedback-only method, we compare the maximum number of teaching signals (feedback + guidance) provided by the human until convergence. As the teacher provides a feedback in every step, the maximum number of teaching signals needed if using only feedback is equal to 391. However, by using unlabeled guidance in addition to feedback, the results show that a teacher needs to provide at most 126 teaching signals (102 feedback +24 guidance signals).

We conclude from these results that using unlabeled guidance signals in addition to evaluative feedback reduces the number of steps (74% less) as well as the number of required teaching signals (67% less).

V. Experiments on the real robot

We compared our model to the feedback-only model with the Baxter Research Robot in both the small and the large state space conditions. We conducted four experiments for each model in each condition which resulted in 16 experiments. All the experiments were performed by one of the authors. A video of one experiment can be found online[4].

In each experiment, all of the six objects are presented one by one to the robot in a specific order. Four different orders were chosen randomly beforehand and the same orders were employed for both models in both conditions. Each experiment ends when each of the six object has been presented two times for the small state space condition and three times for the large state space condition, whether learning converged or not. In all experiments, the teacher provided a feedback for every step. In addition, for the model with guidance, the teacher provided a guidance signal only when the robot did not receive yet any guidance for the given state or when the recorded signal was erroneous.

Figure 6 reports the evolution of the number of provided guidance and negative feedback over time for each condition. The results are averaged over the four experiments. We can see that in the small state space condition, the feedback-only model converged after at most 36 minutes, while the model

[4] http://www.isir.upmc.fr/vid/intRL.mp4

with guidance converged within 17 minutes. In the large state space however, the feedback-only model did not completely converge after an hour of training, while the model with guidance converged after at most 24 minutes.

Table I reports some standard statistics of the experiments such as the training time, the number of steps, number of teaching signals, number of discovered states, number of undesired states and the number of steps spent in undesired states. Undesired states define situations in which the robot is holding the object with the wrong hand, or is holding the object while its descriptors are unknown (this may happen if the robot takes a picture after picking the object). We also report the size of the Q-function measured as the number of state-action pairs for which the algorithm learned a value.

The experimental results are consistent with those obtained in simulation and with the results of [16]. They show that guidance reduces considerably the number of steps and training time (42% less steps for the small state condition and 64% in the large state condition). It is also more efficient by achieving better performance with less interaction (30% less signals for the small state condition and 53% in the large state condition). The robot also discovered less states (respectively 29% and 43% less in each condition), less undesired states (resp. 42% and 65% less) and spent less time in these states (resp. 65% and 85% less).

Finally, the ratio of the number of actions per state for which the robot has learned a value while using only feedback is of 3 actions per state for both conditions. However, by using unlabeled guidance signals the robot learned a value for 7 actions per state in both conditions. This means that our model allows to determine more effectively the Q-value function in less time.

	Small state space		Large state space	
	SVFB	FB	SVFB	FB
training time (mn)	**25**	33	**31**	67
#steps	**135**	235	**166**	470
#feedback	**135**	235	**165**	466
#negative feedback	**42**	135	**33**	296
#guidance	29	**0**	50	**0**
#states	**36**	51	**61**	108
#undesired states	**8**	14	**8**	23
#steps in undesired states	**10**	29	**10**	70
#Q-values	**265**	164	**450**	359

TABLE I: Experiment statistics. The results are averaged over four experiments. Training time, number of steps, number of provided feedback, negative feedback and guidance signals, number of discovered states, number of undesired states, number of steps spent in undesired states and number of learned Q-values.

VI. Discussion

In this section we discuss some aspects of our model to provide some insights about its performance.

We first evaluate the complexity of our model measured as the number of q-values that the robot needs to compute in the worst case scenario. As we already mentioned, the main idea of our model is to learn the task in an alternative state space of reduced complexity and to transfer this knowledge back into the original state space. In our case, the alternative state space is defined over the guidance signals. As each guidance signal corresponds to an action, the size of the guidance state space is equal to the size of the action set (as we do not consider the case where different guidance signals relate to the same action, nor the case where the same guidance signal refers to different actions). If n is the size of the state space and a the size of the action set, learning the task on the original state space comes to compute the values of at most $n \times a$ state-action pairs. However, if we learn the value function on the guidance state space, the complexity of the learning process would be reduced to a^2 state-action values. For example, in the large state space condition of the sorting task, the robot needs to learn only 81 values instead of 7776.

We consider the cost of our method in terms of human load measured as the maximum number of teaching signals needed from the teacher during the learning process. If the robot learns from a predefined reward function, so without evaluative feedback as in SVRL, a teacher needs to provide only guidance for every state, so in total n signals for the Teaching Model to be fully observable. If the robot learns from evaluative feedback instead of task rewards, in the worst case scenario a teacher using guidance needs to provide $n + a^2$ teaching signals instead of $n \times a$ if only using feedback (945 instead of 7776 for the large state space condition).

Our model has some other advantages over using only feedback or using feedback with preprogrammed guidance. On the one hand, feedback is purely reactive as rewards are given after the action is performed. So, it does not allow to anticipate the robot's actions. In our model, however, previously provided rewards are also transferred from the Teaching Model to the Task Model before action is performed (line 8 of Algorithm 1). This explains why it is more efficient in preventing from undesired actions (cf. Table I).

On the other hand, preprogrammed guidance signals are classically used for restricting the choice of possible actions. By doing so, they only inform the robot about the allowed actions but do not inform about the other actions. If we look at the way humans provide guidance, we find that guidance signals are more informative. For example, when a child is about to put his fingers into a power plug and receives the warning from his parent to "not do that!", this signal informs him that this action may be dangerous for him even if he does not actually perform it. This is an important aspect from a safety point of view. In our model, when the robot encounters a new state and receives a guidance signal which action values are already known, all these values are transferred to the task state (line 8 of Algorithm 1). So, the robot is informed about the outcomes of all these actions even if they are never explored in that state. This explains the difference in the number of learned state-action values per state compared to the feedback-only model (cf. Table I).

Another potential advantage of our model is the facility to correct wrong feedback. When using only feedback, if the optimal action is rewarded negatively in a particular state, its value is decreased compared to the other actions.

Consequently, the teacher has to wait until all other actions are explored and rewarded negatively in that state in order to make the robot try again the optimal action and thus to be able to correct the wrong feedback. With our model, however, as several states may share the same action, a wrong feedback given in a particular state has less influence on the value of the associated guidance-action pair. In addition, the teacher will have more frequently the possibility to correct the feedback as the guidance signal may be encountered in many different states.

Finally, if the task changes and has to be relearned, using only feedback requires to restart from scratch by giving once again the new feedback for the entire state-action space ($n \times a$ signals). With our model, once the mapping between guidance and actions has been learned in the Teaching Model, relearning the task is reduced to giving the new guidance signals for each task state (n signals).

Despite all these advantages, our model still has some limitations. First of all, the model relies on discrete guidance signals, which assumes a preprogrammed segmentation of the human gestures. Even with an important category of gestures, the way guidance is provided is still constrained by the gestures the system is able to detect.

Another limitation is the autonomy of the learning process with respect to the human. Using $\gamma = 0$ means that the teacher needs to provide guidance and evaluative feedback in every state, which can be a burden for the teacher in large and complex domains. Using $\gamma \in (0, 1]$ may reduce the dependency on the teacher but may result in unsuccessful learning when learning from evaluative feedback [8], [6]. This is however more suitable when learning from a predefined reward function. Some promising solutions have been proposed for combining both types of reward [7]. But the challenge is still to find an efficient way to design a reward function for real robotic tasks [14].

VII. CONCLUSION AND FUTURE WORK

We presented a model for training a robot using unlabeled guidance signals and evaluative feedback. We demonstrated the advantage of using unlabeled guidance signals over using only evaluative feedback both in simulation and in a real robotic task. We have shown that our model reduces the training time and the number of interactions with respect to using only evaluative feedback.

Although we can consider that our proposed model generalizes action values over different task states by the means of guidance signals, in this paper we did not consider the question of generalization over state features. This is a challenging question as it requires to find the correct generalizations online within a minimum number of steps. Furthermore, we did not verify the robustness of our model against the sparsity and the inconsistency of teaching signals. Also, we considered only the particular case of guidance where there is a direct mapping between guidance signals and intended actions.

In the future, we propose to tackle these questions and to generalize our work to the case where different signals are used for the same action or where the same signal relates to different actions. We also propose to remove the constraint about the segmentation of guidance signals by taking into account continuous signals. Finally, we intend to verify the results obtained in this paper in a broader study by evaluating our model with naive users.

REFERENCES

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469 – 483, 2009.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer vision and image understanding*, 110(3):346–359, 2008.

[3] J. García and F. Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16:1437–1480, 2015.

[4] S. Griffith, K. Subramanian, J. Scholz, C. Isbell, and A. L. Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2625–2633, 2013.

[5] J. Grizou, M. Lopes, and P.-Y. Oudeyer. Robot learning simultaneously a task and how to interpret human instructions. In *Development and Learning and Epigenetic Robotics (ICDL), 2013 IEEE Third Joint International Conference on*, pages 1–8, Aug 2013.

[6] M. Ho, M. L. Littman, F. Cushman, and J. L. Austerweil. Teaching with rewards and punishments: Reinforcement or communication? In *Proceedings of the 37th Annual Meeting of the Cognitive Science Society, CogSci 2015, Pasadena, California, USA, July 22-25, 2015*, 2015.

[7] W. B. Knox and P. Stone. Combining manual feedback with subsequent mdp reward signals for reinforcement learning. In *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, May 2010.

[8] W. B. Knox and P. Stone. Reinforcement learning from human reward: Discounting in episodic tasks. In *RO-MAN, 2012 IEEE*, pages 878–885. IEEE, 2012.

[9] W. B. Knox, P. Stone, and C. Breazeal. Training a robot via human feedback: A case study. In *Social Robotics*, pages 460–470. Springer, 2013.

[10] J. Kober, J. A. D. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, July 2013.

[11] A. León, E. F. Morales, L. Altamirano, and J. R. Ruiz. Teaching a robot to perform task through imitation and on-line feedback. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 549–556. Springer, 2011.

[12] Y. Mohammad and T. Nishida. Learning interaction protocols using augmented baysian networks applied to guided navigation. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 4119–4126. IEEE, 2010.

[13] A. Najar, O. Sigaud, and M. Chetouani. Social-Task learning for HRI. In *Social Robotics*, pages 472–481. Springer, 2015.

[14] A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, pages 663–670, 2000.

[15] P. M. Pilarski, M. R. Dawson, T. Degris, F. Fahimi, J. P. Carey, and R. S. Sutton. Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1–7. IEEE, 2011.

[16] H. B. Suay and S. Chernova. Effect of human guidance and state space size on interactive reinforcement learning. In *RO-MAN, 2011 IEEE*, pages 1–6. IEEE, 2011.

[17] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 1998.

[18] A. L. Thomaz and C. Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *AAAI*, volume 6, pages 1000–1005, 2006.

[19] A. L. Thomaz and C. Breazeal. Teachable robots: Understanding human teaching behavior to build more effective robot learners. *Artificial Intelligence*, 172(6):716–737, 2008.