2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)

Simultaneous Learning of Objective Function and Policy from Interactive Teaching with Corrective Feedback

Carlos Celemin, and Jens Kober

***Abstract*— Some imitation learning approaches rely on Inverse Reinforcement Learning (IRL) methods, to decode and generalize implicit goals given by expert demonstrations. The study of IRL normally has the assumption of available expert demonstrations, which is not always possible. There are Machine Learning methods that allow non-expert teachers to guide robots to learn complex policies, which eventually fills the expert dependencies of IRL. This work introduces an approach for simultaneously teaching robot policies and objective functions from vague human corrective feedback. The main goal is to generalize the insights that a non-expert human teacher provides to the robot, to unseen conditions, without further need for human effort in the complementary training process. We present an experimental validation of the introduced approach for transfer learning of knowledge to scenarios not considered while the non-expert was teaching. Experimental results show that the learned reward functions obtain similar performance in RL processes compared to engineered reward functions used as baseline, both in simulated and real environments.**

I. Introduction

Most of the techniques used for robot learning to execute tasks are based on Reinforcement Learning (RL) or Imitation Learning (IL). IL can be faster in general but is very dependent on the human time, effort, and sometimes high skills. Whereas RL is an autonomous process that requires less human supervision, but is time consuming, which is specially problematic for physical systems. However, RL still requires the effort of an expert user for defining a reward function. Actually this design is most of the times more difficult than providing demonstrations of the task, which has motivated the Inverse Reinforcement Learning (IRL) problem, for learning a reward function that explains the observed behavior, and under which the expert's demonstrations are optimal. [1].

Approaches such as Behavioral Cloning (BC) [2] in which providing demonstrations, and copying the mapping from states to actions is also not always possible. Especially in some problems with complex dynamics, due to the necessity of high skills of the demonstrator, or because mapping the state-action space of the demonstration to the one of the learning agent is sometimes not straightforward [3], [4]. For these problems, there are interactive methods that allow non-expert users to train agents for complex tasks, without the need of recording complete sets of demonstrations in one batch at the beginning of the learning process, but giving the demonstrations, or other kinds of feedback in an incremental fashion, since this is more intuitive for users [5].

All authors are with the Cognitive Robotics Department, TU Delft, Delft, The Netherlands (c.e.celeminpaez, j.kober)@tudelft.nl

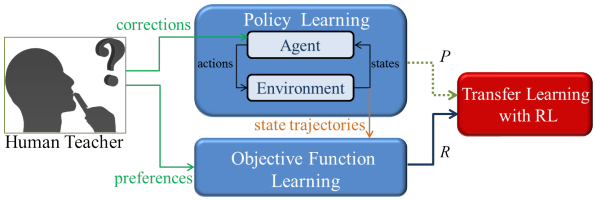


Fig. 1: Scheme of the proposed model-free Simultaneous Objective and Policy Learning approach.

Policies can be trained with DAGGER [6], a method using a data-set of expert demonstrations that is incrementally built online, while the policy is updated after each iteration, with a constantly growing data-set. Although DAGGER has the aforementioned advantage of incremental learning from demonstrations, it still requires the expert demonstrations to be optimal, as BC, in order to reach good performance policies [7]. For instance, BC and DAGGER obtain the same level of performance in block stacking tasks [8].

Learning approaches intended for non-expert users are based on occasional feedback, and mostly based on Interactive RL methods such as [9]–[12], in which the user does not need to be skilled for demonstrating good executions of the task, but just able to evaluate the executed actions with punishment or reward signals. However, these approaches are not preferred by teachers, since user studies have shown that giving feedback in the actions space is rated higher than methods based on evaluating actions or policies, which were found to be less intuitive [13], [14]. In contrast to the previous approaches, non-expert demonstrators can train policies with corrective feedback in the actions domain, with signals that represent relative corrections of the executed actions. The users do not need to know the magnitude of the optimal action, but only the direction of the correction according to their insights about the task, so the policy is incrementally improved through the learning episodes [15]–[19]. Among these methods, the COACH (COrrective Advice Communicated by Humans) algorithms [16], [17] are very data efficient, intended for online training, and have shown to outperform significantly methods based on human reward like TAMER [9], in problems where the users could not successfully demonstrate the task execution. This relative directional corrective feedback has also been used with physical interaction for manipulators [18], [19]. These approaches do not learn controllers but the cost function used to compute trajectories, which is in the scope of IRL.

The reuse of knowledge obtained while teaching a policy has potential for use in transfer learning, thus policies could

be generalized to unseen situations/scenarios during stage of human teaching. In this context, transfer learning could be used to reduce the required human effort/time for learning a robust controller. The previous idea is one of the possible applications of the contribution of this work, since IRL can be used in that regard. In this work, we focus on learning the objective function for complex tasks which cannot be demonstrated by the users, rather, they are trained incrementally online with corrective feedback, i.e., simultaneously learning policy and objective function (Fig. 1). The purpose is to understand the human objective function R^*, that better explains the policy learned incrementally with vague human corrective feedback, in the context of MDPs with unknown transition functions. This is useful for generalizing the obtained knowledge of the task to scenarios with different conditions, where the learned policy is no longer applicable, yet avoiding the effort of the human teacher who would need to provide corrections in all unseen cases. Instead, a human-free transfer learning process is carried out by means of the learned objective function in an RL process. Additionally, the learned policy could be reused as initial point, in cases where the transfer learning keeps the same embodiment (dashed line in Fig. 1). Thus, the applicability of IRL is widened to problems without available expert demonstrations, but in which non expert teachers can train policies.

This paper is organized as follows: Section II briefly introduces works in the context of our proposed approach, and Section III describes the new method and its derivation. The experimental validation is presented in the Section IV, and conclusions are drawn in Section V.

II. Background and Related Work

Learning objective functions from "optimal" demonstrations has been studied with IRL algorithms such as Maximum Margin Planning [20], Maximum Entropy IRL [21], Relative Entropy IRL [22], and Path Integral IRL [23]. They learn reward functions R represented with linear combinations of features Φ as

$$R(s,a) = \sum_{i=1}^{k} w_i \phi_i(s,a) = w^\top \Phi(s,a),$$

wherein s represents the state, a the action, and w the weights vector to be learned. In some cases, the dependency of Φ on the actions is left out to avoid overfitting, especially when embodiment of demonstrations is different from the one of the learning agent, e.g., recording human body movements to be reproduced by a humanoid robot.

A. Learning Objective Functions from Coactive Feedback

Coactive Learning was proposed for understanding and learning objective functions from human preferences through iterative interaction between users and learning agents, and validated with recommendation systems [24]. In this approach, given a context x_t the system presents an object y_t (e.g. ranked list of search results). The utility function $U(x_t, y_t)$ is unknown, but when direct or indirect human feedback is returned with an object $\overline{y_t}$, the utility function is updated such that $U(x_t, \overline{y_t}) > U(x_t, y_t)$. For instance, selection of an object of the list of search result that is not in the top, might mean that it should be the first recommendation. In the context of Markov Decision Processes (MDP), x_t is the state s_t, the object y_t is the action a_t, and the utility function U is equivalent to $R(s,a)$, also a linear combination of features. From now on we switch to the MDP notation.

The feedback given by the user is taken as a correction such that the utility function is optimal with the user's preference $\overline{a}$ as in

$$\overline{a} = \underset{a}{\operatorname{argmax}} \left(R(s,a) \right),$$

therefore the utility function is updated in the direction $\Phi(s_t, \overline{a}_t) - \Phi(s_t, a_t)$. Coactive Learning includes variants of the basic Preference Perceptron algorithm presented in Algorithm 1. This approach has been used to learn the objective function for trajectories executed with robot arms in the Trajectory Preference Perceptron [18], and Online Learning from Physical Human Robot Interaction (pHRI) [19]. In this case, given a state, the robot executes a trajectory, and the user corrects it with physical interaction pushing the robot towards the place s/he prefers the trajectory to pass. The assumption of these methods is that the correction is an informative feedback meaning that the current objective function R is different from the one considered by the human teacher R^*, so it guides how the objective function should be updated.

The reward function obtained in those works is used for planing trajectories instead of learning controllers based on RL. In this work, we propose to extend the application of this concept to learn objective functions from the trajectories observed while interactively teaching an agent to execute a task in the context of MDPs with unknown transition functions.

B. Learning Policies with Human Corrective Feedback

COACH (COrrective Advice Communicated by Humans) [16] is a method in which the user provides corrections to the learning agent whenever considered necessary, as in the mentioned Coactive Learning approaches. However, this method learns the policy rather than the objective function. The main feature is that it works under the assumption of considering the user does not have much domain knowledge of the task, so s/he does not know the exact magnitude of the correction but has insight of the trend of the correction. So, with simple binary corrections h (positive, negative), the policies are incrementally learned. Since the user does not

Algorithm 1 Preference Perceptron

1: Initialize $w_1 \leftarrow 0$
2: **for** $t = 1, 2, \ldots, N$ **do**
3: **observe** state s_t
4: **present** action $a_t \leftarrow \operatorname{argmax}_a \left(w_t^T \Phi(s_t, a) \right)$
5: **obtain** feedback $\overline{a}_t$
6: **update** $w_{t+1} \leftarrow w_t + \Phi(s_t, \overline{a}_t) - \Phi(s_t, a_t)$

know the size of the correction, the policy is updated based on stochastic gradient descent with an assumed constant error magnitude e such that the error is computed $\text{error} = h \cdot e$, $\forall h \in \{-1, 0, 1\}$ according to the human feedback, wherein zero means no correction. Additionally, the corrections are not constrained to kinesthetic interaction and can be given with different interfaces.

This framework shapes two functions parametrized as a linear combination of features $f(s)$. One function is the policy $\pi(s) = \theta^\top f$; the second function $H(s) = \psi^\top f$ learns to predict the human feedback, for computing an adaptive learning rate used in the policy update, weighting the assumed constant error magnitude e (see details in [16]). The parameter vectors θ and ψ are updated to shape the models. As it can be seen, f is the same vector for both the Policy Model $\pi(s)$ and the Human Feedback Model $H(s)$. Algorithm 2 describes the basic COACH for the j-th episode of N time steps, which return the state trajectory $\zeta_j = \{s_1, s_2, ...s_N\}$.

III. Learning Objective Functions from Interactive Learning Policies

This work focuses on integrating the process of learning policies, and simultaneously learning objective functions. Our insight considers that during the incremental process, the policy is constantly improved, so early sub-optimal policies have executed trajectories that should be ranked with lower performances. The assumption of monotonic improvement could be relaxed with the addition of evaluative preference feedback. So we consider in this method that after a training episode is finished, the user assesses the execution compared to the immediately previous episode, so the human preference is $p = 1$ when the trajectory is considered better than the previous one, and $p = -1$ when it is worse.

The derivation of our proposal to learn the objective function while teaching the policy is inspired by pHRI, which is based on maximum entropy assumptions. For the update of the weights w estimate, the observation model is

$$P(\zeta_H|\zeta_R, w) \approx e^{w^T\left(\Phi(\zeta_H)-\Phi(\zeta_R)\right)-\lambda\|\zeta_H-\zeta_R\|^2},$$

Algorithm 2 Basic Structure of a COACH episode

1: **Require:** error magnitude e, human model learning rate β, init. trajectory of the $j-$th episode $\zeta_j \leftarrow \emptyset$
2: **for** $t = 1, 2, \ldots, N$ **do**
3: **observe** state s_t
4: **execute** action $a_t \leftarrow \pi(s_t)$
5: **feedback** human corrective advice h_t
6: **if** $h_t \neq 0$ **then**
7: **update** $H(s_t)$ with $\Delta\psi \leftarrow \beta \cdot (h_t - H(s_t)) \cdot f$
8: $\alpha_t \leftarrow |H(s_t)|$
9: $\text{error}_t \leftarrow h_t \cdot e$
10: **update** $\pi(s_t)$ with $\Delta\theta \leftarrow \alpha_t \cdot \text{error}_t \cdot f$
11: $\zeta_j \leftarrow \zeta_j \cup s_t$

with ζ_R the trajectory the robot plans to execute, and ζ_H the actual trajectory corrected by the human with physical interaction [19]. The prior of the initial estimate of w is

$$P(w) = e^{-\frac{1}{2\alpha}\|w-\hat{w}_0\|^2}. \quad (1)$$

Our work is in the scope of model-free settings, therefore the trajectory ζ_R cannot be planed and known in advance, we only know ζ_H since the policy is receiving human corrections and being updated online. We rely on the assumption that human corrections maximize the robot's reward function of the task execution (episode) with respect to the previous execution, as long as the evaluation p indicates it, otherwise the episode $j-1$ maximizes the reward function with respect to the episode j. Taking this into account, our observation model becomes

$$P(\zeta_j|\zeta_{j-1}, w) \approx e^{w^T\left(\Phi(\zeta_j)-\Phi(\zeta_{j-1})\right)p_j-\lambda\|\zeta_j-\zeta_{j-1}\|^2}. \quad (2)$$

The distribution of w_j is given by $P(\zeta_1, ..., \zeta_j|\zeta_0, ..., \zeta_{j-1}, w)P(w)$. Therefore, using maximum a posteriori (MAP) estimate in (3), with (1) and (2) obtains the MAP in terms of the trajectories in (4).

$$\hat{w}_{j+1} = \underset{w}{\text{argmax}} \left(\sum_{i=1}^{j} \log P(\zeta_i|\zeta_{i-1}, w) + \log P(w) \right) \quad (3)$$

$$\hat{w}_{j+1} \approx \underset{w}{\text{argmax}} \left(\sum_{i=1}^{j} w^T\big(\Phi(\zeta_i) - \Phi(\zeta_{i-1})\big)p_i - \frac{1}{2\alpha}\|w - \hat{w}_0\|^2 \right) \quad (4)$$

Using the gradient with respect to the weights w in (4) obtains the update rule

$$\hat{w}_{j+1} = \hat{w}_0 + \sum_{i=1}^{j} \alpha\big(\Phi(\zeta_i) - \Phi(\zeta_{i-1})\big)p_i. \quad (5)$$

A. Simultaneous Objective And Policy Learning: SOAP

This work has a formulation for learning policies and objective functions simultaneously, called SOAP. Nevertheless, the policy is updated online after every time step of human corrections, whereas the reward function is updated offline after finishing every episode. The last remark is different from the approach of pHRI, because in that method the human corrections obtain a new cost function in the next time step, which is used to re-plan the trajectory immediately. This is not a problem in this context, since we aim at MDPs with unknown transition function, wherein planning is not possible. Moreover, in this context the obtained objective function is not needed during the training episodes, instead, it is to be used later within a RL process for indirect imitation of the taught behavior.

In Algorithm 3, the steps that integrate the incremental process of learning policies and objective functions with vague human corrections is listed. A first episode of teaching with COACH (Algorithm 2) is run in line 3 to obtain the trajectory ζ_0, which does not have another trajectory to be compared with. There is a loop in lines 3-7 that is repeated

until the human teacher considers the learning policy is good enough at executing the intended task. Thus, every iteration j, a new trajectory ζ_j is recorded (line 4), and evaluated by the human teacher in contrast to the previous one (line 5). Then, the weights of the objective function are updated with the new data based on (5) (line 6), and finally the teacher decides whether to continue or to stop, in case the policy is reaching her/his expectations.

B. Extension to Relative Entropy IRL using preferences

Although we have introduced a method for learning the objective function with data obtained from an interactive process of teaching policies, the main objective is to show, that the observed trajectories during teaching a policy with COACH posses information regarding what is desired about the task and what is not. We argue that other model-free IRL methods could be used with this data, and that they are not limited to scenarios with real expert demonstrations. The update rule in line 6 could be modified to a different approach, or indeed, the computation of w can be carried out after the "while" loop in Algorithm 3, directly from the batch of recorded trajectories ζ. For instance, Relative Entropy IRL (REIRL) [22], could be implemented like in [25] considering the set of non expert demonstrations formed by ζ_j, $\forall j \in \{1, 2, ..., E-1\}$, and ζ_E the expert trajectory, wherein E is the number of COACH episodes run for teaching the policy.

Additionally, we consider including the information of the human preference evaluation p_j, for modifying the reference distribution Q [22], which is intended for encoding prior preferences and constraints [25]. Then, we assume to give a linear increment on the probability $Q(\zeta_j)$ with respect to $Q(\zeta_{j-1})$, according to the selected preference p_j. Since the first trajectory ζ_0 is not compared with a previous one, we assume $p_0 = 0$. Then a ranking of the trajectories Rank_j is computed in (6) with the strong assumption that every iteration, the performance of the policy is modified with a constant rate.

$$\forall k \in \{0, ..., E-1\} : \mathrm{Rank}_j = \sum_{i=0}^{j} p_i + 1 - \min_k \sum_{i=0}^{k} p_i \quad (6)$$

Therefore, in (7) the mapping of the ranking into a prior in the reference distribution Q is given.

$$Q(\zeta_j) = \frac{\mathrm{Rank}_j}{\sum_{i=0}^{E-1} \mathrm{Rank}_i} \quad (7)$$

Algorithm 3 Simultaneous Objective and Policy learning

1: Initialize $w_1 \leftarrow 0$
2: **observe** trajectory $\zeta_0 \leftarrow \mathrm{COACHepisode}$
3: **while** Performance $\neq$ ok **do**
4: **observe** trajectory $\zeta_j \leftarrow \mathrm{COACHepisode}$
5: **query** human preference $p_j \leftarrow \mathrm{HumanPreference}$
6: **update** $w_{j+1} \leftarrow w_j + \alpha\big(\Phi(\zeta_j) - \Phi(\zeta_j - 1)\big)p_j$
7: **query** Performance $\leftarrow \mathrm{HumanGreenLight}$
8: $j \leftarrow j + 1$

With this prior, REIRL gives priority to the best non expert trajectories, which makes the algorithm focus on the details, that allow discriminating the expert trajectories with respect to the best non expert ones.

IV. Experimental Results

The validation of our proposal of simultaneously learning policies and objective functions was carried out with two different tasks, one in a simulated environment, while the other is with a real robot arm. The proposed approach is compared to the Relative Entropy IRL algorithm (REIRL), and our naive variation that includes the human preferences in the reference distribution, presented in Section III-B, which we call REIRLp.

Additionally, for each problem, we also compare the performance of learning the task based on a "real" reward function as a baseline, which is the original reward function defined for the task. This is in order to observe the performance of the learned objective functions, with respect to engineered ones. We do not expect to necessarily learn faster or achieve higher outcomes, but to observe reasonable learning processes based on the learned reward functions, whose reach is compared by a baseline reward function.

A. Transfer Learning in the Cart-Pole setting

Our approach is validated with this simulated environment [26]. The goal of the task is to maximize the number of time steps the pole (attached on top of a cart) is balanced, with a maximum limit of 5000 steps. The state vector is composed by the position and velocity of the cart, angle and angular velocity of the pole $[x, \dot{x}, \theta, \dot{\theta}]$.

We carried out the process of simultaneous learning a policy and an objective function, with episodes that had always the same initial conditions, so it was run the SOAP approach once with initial state vector $[0, 0, 0, 0.01]$, then the learned w vector was used to initialize the w_1 in a second run of Algorithm 3, setting initial state $[0, 0, 0, -0.6]$. The two initial state vectors have only non zero angular velocities, although those are low magnitudes. Having a fixed initial condition helps the human teacher to learn quickly to predict what happens at the beginning of the episode, so s/he can predict how to correct the policy in the very first time steps, such that the episode is not finished very soon without much advice of the teacher. Fixing the initial conditions instead of using random initialization prevents the learning process to obtain a robust controller that is not sensitive to those initial states. Nevertheless, the purpose of our approach is intended for scenarios like this, so we make the human teacher train a controller in a couple of easy situations, and then the obtained knowledge (policy and objective function) is used to generalize onto unseen situations, without the need of further human effort.

The obtained knowledge is used to first validate how an RL agent can learn to execute the task, in the same conditions observed during the COACH episodes. Secondly, that knowledge is used for transfer learning, such that an RL agent can obtain a policy for new situations, wherein the

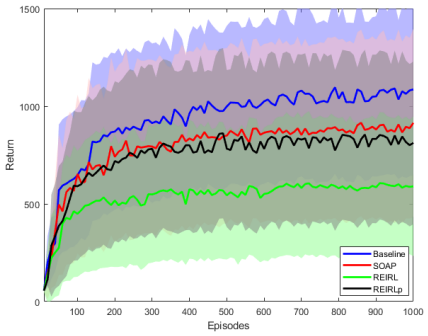


Fig. 2: Average learning curves for the same initial conditions set for the COACH episodes.

policy trained with COACH does not perform well. For every studied case, we carried out 50 runs (1000 episodes each) of the RL process for every considered objective function, i.e., the obtained with SOAP, REIRL, its slight variation REIRLp, and the baseline reward function. We present the statistical results with the average learning curve of every evaluated case and its standard deviation.

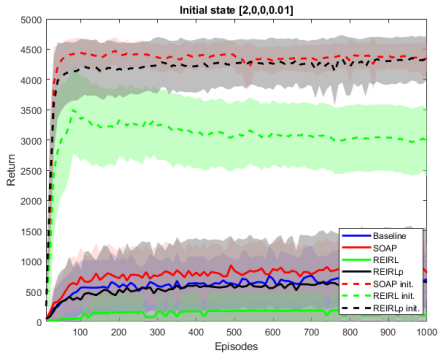
Fig. 2 depicts the evolution of the learning process under the different objective functions. The performance of the objective functions based on the two introduced approaches outperformed REIRL, reaching around 80% of the outcome obtained with the baseline reward function. SOAP obtains a performance slightly higher than REIRLp.

The transfer learning was validated in two main cases: (i) to unknown initial conditions; (ii) to environments with changed dynamics.

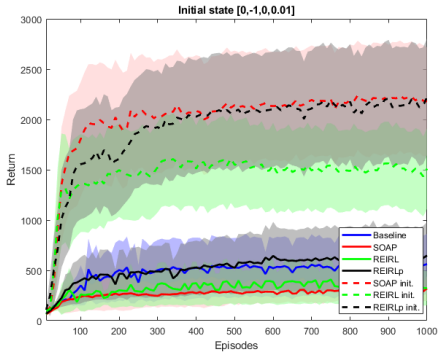
1) Transfer learning to unseen initial conditions: For the case of unseen initial conditions, Fig. 3 depicts the results of two scenarios in which the initial state is different from the one observed while the human teacher was training the agent. One case started with a different initial position of the cart, and the second started with different initial velocity of the cart. Additionally for these scenarios, we also executed RL with policies initialized with the one trained by the human teacher. For the cases wherein the policy was learned from scratch, the baseline reward was slightly outperformed by SOAP in the case of the new initial cart position, and by REIRLp in the case of the new initial cart velocity.

More interesting results are obtained when in the process of transfer learning, it not only uses the objective function, but also the actual policy learned with the human corrections. In Fig. 3 the dashed lines show how RL performs with reuse of the taught policy. In the very beginning, the original policy has a poor performance because it cannot balance the pole at all in the first steps of the episode, however the rate of improvement is very high, reaching significantly better performance levels than obtained by learning from scratch. In the two experiment scenarios, SOAP and REIRLp obtained very similar progress, which is more than 40% higher than the achieved with regular REIRL.

2) Transfer learning to environments with changed dynamics: For this case, we explored two scenarios wherein



(a) Different initial cart position.



(b) Different initial cart velocity.

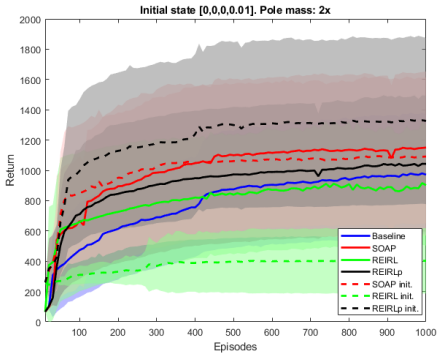
Fig. 3: Transfer learning, average learning curves for cases of different initial conditions.

either the mass of the cart or the pole was doubled. In these cases the policy trained by the human teacher on the original dynamics could not prevent the pole from falling at the beginning of the episode, so further learning was necessary.

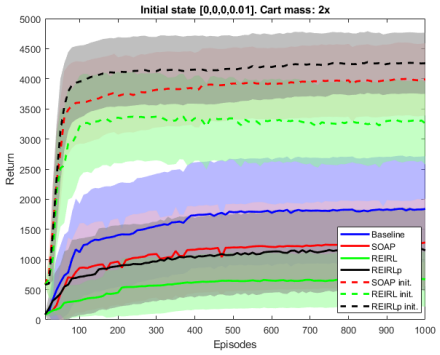
Fig. 4 depicts the learning progress for the situation with the heavier pole in (a), and the heavier cart in (b). When the mass of the pole was twice the original, the approaches introduced in this work learned better policies than the RL using the baseline function, and only REIRL had lower performance. The environment with heavier cart achieves similar results, however the baseline reward function was significantly outperformed when the policy learned with COACH was used to initialize the RL process. For this case, RERIL was outperformed by our proposed SOAP and the variant RERILp.

B. Visual Servoing for tracking a moving object

This problem of Visual Servoing (VS) is used as a proof of concept of the whole process introduced in this paper. In this task, a 7 DoF lightweight robot arm KUKA iiwa is used to track a moving object, with a camera installed in the end-effector. The policy of the robot has to compute the joint velocities, based on the observed position and velocity of the object in the image with respect to the center of the same frame. This distance is normalized such that the extreme coordinates are 1 and -1 in both x and y axes, i.e., when the object is in the margin of the camera frame. Since the task is to track the object, the objective is to minimize the



(a) pole with double mass.



(b) cart with double mass.

Fig. 4: Transfer learning, average learning curves for cases of different dynamics of the environment.

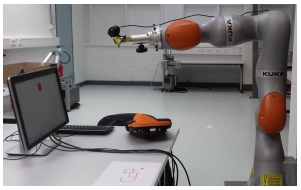
distance between the center of the camera, and the coordinate in the image of the moving object, for all time steps. Thus, the baseline reward function is set as the negative of the Euclidean distance between the object and the center of the camera image

$$r = -100\sqrt{x^2 + y^2}. \tag{8}$$

In order to make the proof of concept of indirect imitation learning based on the introduced methods, we carried out a process of transfer learning, wherein the policy trained with the corrective feedback uses 2 joints that are locked for the settings with new conditions, so the policy trained by the user is no longer valid. Therefore, the RL process needs to search how to imitate the taught behavior with different actions, based on the learned objective function.

For the interactive stage of teaching the policy, and also for the stage of RL, the episode is set with a fixed duration (600 time steps at 30Hz), in which the robot is in front of a screen that displays the moving object. During the episode the object follows a fixed trajectory which includes three via points, where the object stops completely for few seconds, so the policy learns to track and stabilize in the object position. In Fig. 5(a) the setup of the robot with the camera in front of the screen is shown, while in Fig. 5(b) is shown the image obtained by the camera.

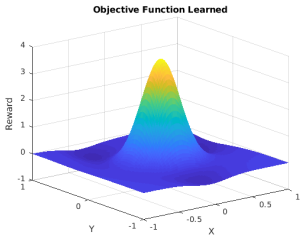
In order to have same observability as the baseline reward function in (8), the obtained reward functions are based on basis functions $\Phi(\zeta)$ that map only position to the features space, without considering velocity. The landscape of the obtained reward function is presented in Fig. 6, wherein it



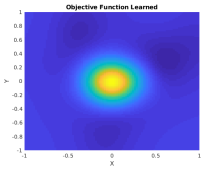
(a) Setup for VS task.

(b) Image observed by the robot.

Fig. 5: Visual Servoing (VS) setup.



(a) Reward landscape.



(b) Top view.

Fig. 6: Learned reward function.

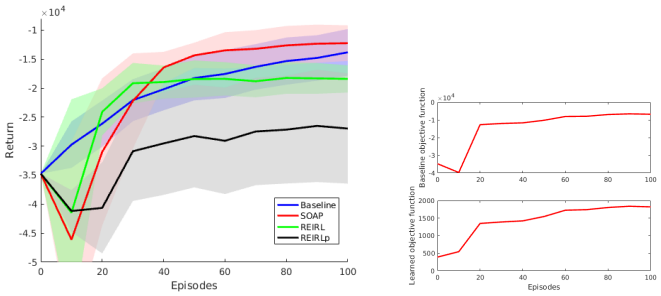
is possible to see, that it gives a high reward to situations where the position of the object is in the center of the camera frame, whereas there are 3 subtle valleys that consider punishing coordinates that are far from the center. These valleys correspond to the three situations of the episode when the moving object stops for a while, so in the very early COACH episodes those areas are visited with more frequency because the policy is still too slow to track the object. In other words, the obtained objective function gives the strongest punishment, to the states that were visited with the policy in the very early COACH episodes.

The reward function learned from the human corrections was computed with 2 runs of our approach SOAP. Therefore, for the second run of Algorithm 3, the vector w_1 is initialized with the vector obtained in the first run. The first run required 17 COACH episodes, while the second one needed 23.

For the comparative study, we carried out 20 runs (100 episodes each) of the RL process for every considered objective function, i.e., the ones obtained with SOAP, REIRL, its slight variation REIRLp, and the baseline reward function. In Fig. 7(a) the mean and standard deviation of the learning curves for the transfer learning process based on RL are depicted. For this case, SOAP obtains the highest average return according to (8), which is higher than the one obtained with the baseline function. Some RL results are in the link[1].

The learning curve of SOAP shows how in the first episodes, the performance of the policy is indeed decreased, and later improved with high rate. We associate this events to the local maxima of the reward function depicted in Fig. 7(a), wherein the predicted reward for unseen states during teaching (e.g. the corners of the frame) is higher than the reward given in the valleys, which may lead the policy to learn to move towards the corners. This means that despite the fact that the return shown in Fig. 7(a) is decreased in early episodes, according to the objective function used by the RL,

[1]https://youtu.be/sKEZLH6Lldk



(a) (b)

Fig. 7: (a) Average learning curves for the transfer learning of the VS task. Return computed with the baseline reward function (8). (b) Performance during a run of RL based on an objective function learned with SOAP. On top, the performance measured with the baseline reward; on bottom the performance observed by the RL agent, based on SOAP

the performance may be increasing due to those erroneous local maxima. For instance, in Fig. 7(b) the learning curve of an RL agent based on SOAP is plotted, the executed episodes were evaluated with both the learned reward function used in the RL, and the baseline one. It is possible to observe that the learned reward function describes proportionally in detail almost the same improvements explained by the baseline function, excepting the first episodes due to the local maxima shown in Fig. 6.

V. Conclusions

This work has introduced an approach of learning objective functions from non-expert's feedback, that explains the goal of a taught policy, by means of reuse of the observed experience during the interactive teaching process with corrective feedback. We argue, that one of the most interesting uses of the knowledge obtained from a teacher with this approach, could be the generalization towards unseen conditions during training.

Simultaneous Objective and Policy learning (SOAP) is based on principles used in model-based and model-free approaches like Coactive Learning, pHRI, and COACH. The proposed method uses the features that are useful to rank and discriminate pairs of consecutive trajectories, which are observed during training. The proposed methodology was tested with transfer learning proofs of concept, in simulated and real systems. The tests were carried out in situations wherein the learned policy could and could not be reused (Fig. 1). In the latter case it is due to changes of action domains. Results show that the contributed approach can obtain objective functions that perform similarly to encoded engineered reward functions within RL processes. Indeed, in the visual servoing task, the learned objective function seems to be more informative than the baseline, since the corresponding learning curve has better convergence.

Moreover, this paper not only argues about the potential of the proposed SOAP, we consider that the trajectories recorded during interactive learning processes like COACH can be used with other IRL algorithms, e.g., as shown with RERIL. Therefore, the applicability of model free IRL is extended to domains lacking of expert demonstrations.

References

[1] A. Y. Ng, S. J. Russell *et al.*, "Algorithms for inverse reinforcement learning." in *Int. Conf. Machine Learning*, 2000.

[2] R. S. Michalski, I. Bratko, and A. Bratko, *Machine learning and data mining; methods and applications*. John Wiley & Sons, Inc., 1998.

[3] 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.

[4] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz, "Keyframe-based learning from demonstration," *Int. J. Social Robotics*, vol. 4, no. 4, pp. 343–355, 2012.

[5] A. Jain, B. Wojcik, T. Joachims, and A. Saxena, "Learning trajectory preferences for manipulators via iterative improvement," in *Advances in Neural Information Processing Systems*, 2013.

[6] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Int. Conf. AI and Statistics*, 2011.

[7] B. Kim, A.-m. Farahmand, J. Pineau, and D. Precup, "Learning from limited demonstrations," in *Advances in Neural Information Processing Systems*, 2013.

[8] Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, "One-shot imitation learning," in *Advances in Neural Information Processing Systems*, 2017.

[9] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: The TAMER framework," in *Int. Conf. Knowledge Capture*, 2009.

[10] H. B. Suay and S. Chernova, "Effect of human guidance and state space size on interactive reinforcement learning," in *Int. Symp. Robot and Human Interactive Communication*, 2011.

[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 *Iberoamerican Congress on Pattern Recognition*, 2011.

[12] G. Warnell, N. Waytowich, V. Lawhern, and P. Stone, "Deep TAMER: Interactive agent shaping in high-dimensional state spaces," 2017, arXiv:1709.10163 [cs.AI].

[13] H. B. Suay, R. Toris, and S. Chernova, "A practical comparison of three robot learning from demonstration algorithm," *Int. J. Social Robotics*, vol. 4, no. 4, pp. 319–330, 2012.

[14] A. L. Thomaz, G. Hoffman, and C. Breazeal, "Reinforcement learning with human teachers: Understanding how people want to teach robots," in *Int. Symp. Robot and Human Interactive Communication*, 2006.

[15] B. D. Argall, B. Browning, and M. Veloso, "Learning robot motion control with demonstration and advice-operators," in *Int. Conf. Intelligent Robots and Systems*, 2008.

[16] C. Celemin and J. Ruiz-del Solar, "An interactive framework for learning continuous actions policies based on corrective feedback," *J. Intelligent & Robotic Systems*, pp. 1–21, 2018.

[17] R. Perez, C. Celemin, J. Ruiz-del Solar, and J. Kober, "Continuous control for high-dimensional state spaces: An interactive learning approach," in *Int. Conf. Robotics and Automation*, 2019.

[18] A. Jain, S. Sharma, T. Joachims, and A. Saxena, "Learning preferences for manipulation tasks from online coactive feedback," *Int. J. Robotics Research*, vol. 34, no. 10, pp. 1296–1313, 2015.

[19] A. Bajcsy, D. P. Losey, M. K. OMalley, and A. D. Dragan, "Learning robot objectives from physical human interaction," in *Conf. Robot Learning*, 2017.

[20] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, "Maximum margin planning," in *Int. Conf. Machine Learning*, 2006.

[21] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning." in *AAAI*, 2008.

[22] A. Boularias, J. Kober, and J. Peters, "Relative entropy inverse reinforcement learning," in *Int. Conf. AI and Statistics*, 2011.

[23] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, "Learning objective functions for manipulation," in *Int. Conf. Robotics and Automation*, 2013.

[24] P. Shivaswamy and T. Joachims, "Coactive learning," *J. Artificial Intelligence Research*, vol. 53, pp. 1–40, 2015.

[25] K. Muelling, A. Boularias, B. Mohler, B. Schölkopf, and J. Peters, "Learning strategies in table tennis using inverse reinforcement learning," *Biological Cybernetics*, vol. 108, no. 5, pp. 603–619, 2014.

[26] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.