A fast hybrid reinforcement learning framework with human corrective feedback

Celemin, Carlos; Ruiz-del-Solar, Javier; Kober, Jens

DOI
10.1007/s10514-018-9786-6

Publication date
2018

Document Version
Final published version

Published in
Autonomous Robots

Citation (APA)
Celemin, C., Ruiz-del-Solar, J., & Kober, J. (2018). A fast hybrid reinforcement learning framework with human corrective feedback. *Autonomous Robots*, *43 (2019)*(5), 1173-1186. https://doi.org/10.1007/s10514-018-9786-6

Autonomous Robots
https://doi.org/10.1007/s10514-018-9786-6

A fast hybrid reinforcement learning framework with human corrective feedback

Carlos Celemin[1,2] · Javier Ruiz-del-Solar[2] · Jens Kober[1]

Received: 21 July 2017 / Accepted: 10 July 2018

Abstract

Reinforcement Learning agents can be supported by feedback from human teachers in the learning loop that guides the learning process. In this work we propose two hybrid strategies of Policy Search Reinforcement Learning and Interactive Machine Learning that benefit from both sources of information, the cost function and the human corrective feedback, for accelerating the convergence and improving the final performance of the learning process. Experiments with simulated and real systems of balancing tasks and a 3 DoF robot arm validate the advantages of the proposed learning strategies: (i) they speed up the convergence of the learning process between 3 and 30 times, saving considerable time during the agent adaptation, and (ii) they allow including non-expert feedback because they have low sensibility to erroneous human advice.

Keywords Reinforcement learning · Learning from demonstration · Policy search · Interactive machine learning

1 Introduction

An important issue of Reinforcement Learning (RL) methods is the relative long training time of systems/controllers to be used in complex/dynamic environments, which can be a limitation for their application in robots interacting in the real-world. This shortcoming can be addressed with the support of human users who participate/collaborate in the learning process. Thus, Learning from Demonstration (LfD) and RL can be sequentially combined for learning skills autonomously from an initial policy that could be obtained by human teachers' demonstrations (Atkeson and Schaal 1997; Kober et al. 2012).

There are some approaches that combine RL with human reinforcements (Thomaz and Breazeal 2006; Tenorio-Gonzalez et al. 2010; Knox and Stone 2012). This combination takes advantage of the user's knowledge for speeding up the learning process while keeping the convergence properties of RL algorithms.

Thus, combining RL with interactive machine learning strategies might be a synergetic relationship, in which the learning processes are sped up as RL benefits from the human knowledge of the task. The RL part also provides more stability and robustness to the interactive framework, which is susceptible to the inherent occasional erroneous feedback associated to human teachers (human are not perfect and prone to fail in repetitive tasks).

In this context, we postulate that the use of reinforcement learning with human feedback will allow addressing important requirements of robotics applications. Since Policy Search (PS) RL seems to be more appropriate than value based RL for facing the challenges of robot RL (Deisenroth et al. 2013), this paper proposes the use of learning methods based on PS techniques that additionally make use of available human knowledge for reducing the learning time, which is one of the main constraints of classical robot learning applications. Corrective feedback advised by human teachers is used in the introduced approach, similarly to the mentioned hybrid learning systems based on RL and human reinforcements. In the proposed approach, human knowledge

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10514-018-9786-6) contains supplementary material, which is available to authorized users.

✉ Carlos Celemin
carlos.celemin@ing.uchile.cl

Javier Ruiz-del-Solar
jruiz@ing.uchile.cl

Jens Kober
j.kober@tudelft.nl

[1] Cognitive Robotics Department, Delft University of Technology, Delft, The Netherlands

[2] Electrical Engineering Department and Advanced Mining Technology Center, University of Chile, Santiago, Chile

Published online: 09 August 2018

is provided to the PS learning agents with corrective advice using the COACH algorithm (Celemin and Ruiz-del Solar 2015), which has outperformed some pure autonomous RL agents and pure interactive learning agents based on human reinforcements, and demonstrated to be useful in some continuous actions problems such as the balancing of cart-pole problem, bike balancing, and also navigation for humanoid robots (Celemin and Ruiz-del Solar 2018).

In the presented hybrid approaches, the corrective feedback given in the action domain by the human teachers speeds up the learning process dramatically, especially during the early stage. On the other hand, the reward based feedback encoded in the cost function plays a more important role for refining the policy when the human perception and understanding capabilities no longer benefit the policy improvement. This is supported by the results of the experiments with 2 simulated balancing environments (cart-pole, and inverted pendulum swing-up) and 3 real systems (inverted pendulum swing-up, inverted wedge, and learning an inverse kinematics model for a 3DoF robot arm). In the validation, the performance of pure PS, and pure COACH is compared against the proposed hybrid approaches. The results show that the hybrid approaches learn faster and obtain better policies than the original methods. The considerable time reductions render learning on real physical systems feasible.

Some experiments show that the RL component provide robustness to noisy and erroneous corrections of the human teacher, while the pure human corrective advice based method is more sensitive to these situations that are always associated to humans. Additionally, in the last part of the experimental evaluation of this work, experiments show that using these hybrid interactive RL approaches, non-expert users can achieve policies with better performance than obtained while they try to operate and execute the tasks directly.

This paper is organized as follows: Sect. 2 provides a brief overview of the kind of PS algorithms explored in the work, followed with the Interactive Machine Learning (IML) strategy based on human continuous corrective feedback to be combined with PS. Section 3 presents the algorithmic formulation of the proposed hybrid learning approaches, based on PS and human corrective feedback. Section 4 presents experimental results, and Sect. 5 presents conclusions and future work.

2 Background and related work

Both autonomous and interactive machine learning strategies have shown to reduce either the workload of the user or the time required for the convergence. For instance, robotic arms can already learn autonomously to solve manipulation tasks based on RL, PS, and deep neural networks directly from raw images without any pre-processing (Levine et al. 2015). Also using RL, several industrial arms are able to simultaneously learn by trial and error to execute grasping tasks directly from the camera images without the necessity of executing calibration processes by the user (Levine et al. 2016).

IML techniques involve users that interact with the agents for adapting the policies (Chernova and Thomaz 2014; Argall et al. 2009). For instance, actions and state representations are learned from user demonstrations for learning manipulation tasks in Rozo et al. (2013), and the problems of *what* and *how to imitate?* are addressed. There are learning strategies that learn from occasional human feedback that could be either corrective in the actions space, or evaluative, as mentioned below.

Corrective feedback has been used in Argall et al. (2008, 2011), wherein policies for continuous action problems are learned from human corrective advice; this kind of feedback also showed to be faster than critic-only RL algorithms for the reported experiments, even when the users were non-experts Celemin and Ruiz-del Solar (2015, 2018).

Evaluative feedback has been reported in applications of RL with human reinforcements, for instance, in Pilarski et al. (2011) a myoelectric controller of a robotic arm is learned through a RL approach that employs the reinforcement from a human teacher instead of an encoded reward function. An object sorting task is learned from human rewards and guidance in Najar et al. (2016). *Human preferences* based approaches use another type of evaluative feedback. Here the teacher iteratively chooses the preferred execution among two policies demonstrated by the agent, leading to good results in simple tasks (Akrour et al. 2011, 2014). This approach has also been applied to learn complex simulated tasks with deep neural networks (Christiano et al. 2017), and applied to manipulation tasks with real robots (Jain et al. 2013).

One of the best known approaches of sequential combination of interactive and autonomous learning is *Apprenticeship Learning* (Abbeel and Ng 2004). This approach has a first stage, wherein a human begins controlling the agent. Then, in a second stage, a reward function is derived with Inverse Reinforcement Learning (IRL) (Ng and Russell 2000; Zhifei and Joo 2012) from the collected demonstrations. Finally, this reward function is used in a standard RL process.

A brief overview of the algorithms used in the proposed hybrid learning scheme is presented below.

2.1 Policy search

Policy Search (PS) is a branch of RL where parametrized policies are learned directly in the parameter space, based on the cost given by the reward function. Thus, PS methods do not learn a value function as most of RL algorithms do. For this reason, PS has advantages with respect to value function based RL in robotic applications, because computing the value function requires data from the complete state-action space. Additionally, the use of parametrized policies reduces the search space, which is important in applications with physical limitations as is usually the case when learning with robots (Deisenroth et al. 2013).

Moreover, in robotic applications PS is a better choice compared to value-based methods due to the properties of scalability and stability of the convergence (Kober et al. 2013). That is, a small change of the policy may lead to a big change of the value function, that can again produce a big modification of the policy. This instability is not necessarily a problem for finding global optima after exhaustive training in simulated environments, however with real robots a smooth and fast convergence is desired.

The general structure of a PS method is presented in Algorithm 1, which includes three main steps: exploration, evaluation, and updating. The exploration step creates samples of the current policy for executing each roll-out or episode. This process can be step-based or episode-based, depending on whether the exploration elements are changing through the time steps or are constant during each episode, respectively. In the evaluation step, the quality of the executed roll-outs is examined, and the strategies can also be step-based or episode-based, depending on whether the assessment is over every single action or over the parameter vector. The update step uses the evaluation of the roll-outs to compute the new parameters of the policy. This update can be based on policy gradients, expectation-maximization, information theoretic, or stochastic optimization approaches. During the last years, several PS algorithms have been proposed and evaluated using different strategies in each of the three steps. Nevertheless, the most suitable method to be used depends on the specific application being addressed. In this work we use episode-based methods with stochastic optimization update strategies that can be considered black-box optimizers, due to the good combination of simplicity and good learning convergence (Stulp and Sigaud 2012b, 2013; Heidrich-Meisner and Igel 2008). Its simplicity makes it easier to combine PS with other learning approaches. In Algorithm 2, the episode-based model free PS scheme used in this work is presented. In the exploration step, a set of noisy samples of the current policy parameter vector is generated. In the evaluation step, a global cost R of each *m-th* roll-out corresponding to the *m-th* sample of the policy parameter vector is measured. In the third step, the policy is updated using the exploration samples and their respective evaluations. In the next paragraphs three episode-based black-box methods used for policy improvement are described.

Algorithm 1: Model Free Policy Search.

```
repeat
    Explore: Execute M roll-outs using policy $\pi_k$
    Evaluate: Obtain outcomes of trajectories or actions
    Update: Compute $\pi_{k+1}$ given the roll-outs and evaluations
until Policy converges $\pi_{k+1} \approx \pi_k$
```

Algorithm 2: Policy Search with episode-based policy evaluation.

```
repeat
    Explore: sample parameter vector
        $\theta^{[m]} \sim \pi_k(\theta), \quad m = 1 \ldots M$
    Evaluate: cost of each roll-out $R^{[m]} = \sum_{t=0}^{T} r_t^{[m]}$
    Update: Compute new policy parameters using
        $\{\theta^{[m]}, R^{[m]}\}$
until Policy converges $\pi_{k+1} \approx \pi_k$
```

2.1.1 Cross-entropy method

The Cross-Entropy Method (CEM) for policy search was proposed in Mannor et al. (2003), and more recently has been used by Busoniu et al. (2011) Stulp and Sigaud (2012a) for learning problems of discrete and continuous actions. Using this approach, the method creates M samples of the current policy for the exploration step, according to a normal distribution as:

$$\theta^{[m]} \sim \mathcal{N}(\theta_k, \Sigma_k), m = 1, \ldots, M \tag{1}$$

where θ_k is the policy vector at the *k-th* iteration and Σ_k the covariance matrix. The cost function must be minimized and return a scalar that represents the performance index. The update step is executed first by sorting the samples in ascending order with respect to the cost function $R^{[m]}$. Then, the normal distribution is updated as:

$$\theta_{k+1} = \sum_{m=1}^{M_e} \frac{1}{M_e} \theta^{[m]} \tag{2}$$

$$\Sigma_{k+1} = \sum_{m=1}^{M_e} \frac{1}{M_e} (\theta^{[m]} - \theta_{k+1})(\theta^{[m]} - \theta_{k+1})^T \tag{3}$$

taking into account only the first M_e “elite” samples with the lowest costs, i.e., $M_e < M$.

2.1.2 Episode-based PI^2

The Path Integral Policy Improvement (PI^2) is a PS method formulated from the principles of stochastic optimal control, and is based on probability weighted average for updating the policy parameters, specially for learning in trajectory control problems (Theodorou et al. 2010). In Stulp and Sigaud (2012a) PI^2 with covariance matrix adaptation (PI^2-CMA) is presented, which combines the standard PI^2 with a computation borrowed from the CEM to automatically adapt the exploration term. An episode-based version of PI^2 that can be classified as Black-Box optimization called PI^{BB} is presented in Stulp and Sigaud (2012b), obtaining interesting results with respect to the original version and other variants.

This method executes the exploration step like the CEM in (1), however, the covariance Σ is not updated by the method itself, but rather the size of this exploration term has to be adapted with an additional strategy that is not part of the algorithm, usually this covariance is diminished through the time. The evaluation for obtaining $R^{[m]}$ has the same restrictions as for CEM, but before the update, the evaluation set is normalized between 0 (the best roll-out) and 1 (the worst). Then, the normalized costs are mapped to a probability distribution $P^{[m]}$ as:

$$P^{[m]} = \frac{e^{-\frac{1}{\lambda}R^{[m]}}}{\sum_{n=1}^{M} e^{-\frac{1}{\lambda}R^{[n]}}} \tag{4}$$

with λ the temperature parameter. Finally, the update step is carried out with the probability-weighted average using the parameter vector of every roll-out and its respective probability $P^{[m]}$:

$$\theta_{k+1} = \sum_{m=1}^{M} P^{[m]}\theta^{[m]}. \tag{5}$$

2.1.3 Episode-based PI^2 with covariance matrix adaptation

In the present work, we propose to extend the PI^2-CMA presented in Stulp and Sigaud (2012a) to PI^{BB}. This extension is intended to incorporate an automatic adaptation of the exploration term in PI^{BB}, with a similar strategy as the one used in (3) for CEM. This alternative strategy can be also seen as a combination of PI^{BB} and CEM. This extension uses the same algorithm employed by PI^{BB}, but it updates Σ according to

$$\Sigma_{k+1} = \sum_{m=1}^{M} P^{[m]}(\theta^{[m]} - \theta_{k+1})(\theta^{[m]} - \theta_{k+1})^T \tag{6}$$

instead of (3), based on the probability computed using (4). We refer to this approach as PI^{BB}-CMA.

The three presented PS algorithms along with the original PI^2 and the PI^2-CMA have slight differences that are summarized in Table 1. The "Covariance Adaptation" field indicates whether the algorithms have covariance adaptation for the exploration distribution or not; the field "Exploration" shows the strategy for disturbing the parameters vector for exploration which can be episode based, or time-step based; the column of "Evaluation" shows the kind of evaluation that the algorithm has, for episode based evaluations it is used a "trajectory cost" which is a scalar index, whereas for time step based evaluations the "cost to go" is computed for every time step; all the listed algorithms update the policy with probability weighted averaging, so the corresponding field indicates the technique the algorithm uses to compute the probability for updating the policy. For instance, CEM takes the best M_e roll-outs with associated uniform probability $1/M_e$ used in (2) and (3), the rest of the algorithms use normalized exponentiation of the cost computed with (4).

Additionally, the proposed methods of this work are included in this table, but the details are presented in the next section.

2.2 Learning from human corrective advice

COrrective Advice Communicated by Humans (COACH) was proposed for training agents interactively during task execution (Celemin and Ruiz-del Solar 2015). In this framework, the state space is represented with the vector $s \in S$. The policy $P : S \to \mathbb{R}^D$ maps directly from states to continuous actions, and occasionally human teachers suggest corrections for the performed actions immediately after their execution with vague binary signals $h \in [-1, 1]$ (increase, decrease respectively). The advice is a relative change of the action's magnitude, which is used for updating the policy with Stochastic Gradient Descent (SGD). The binary signals are "to increase" or "to decrease" the executed action, and could be independently given for each of the D degrees of freedom that compose the action space A. The frame-

Table 1 PS comparison

Algorithm	Covariance adaptation	Exploration	Evaluation	Probability for weighted averaging
CEM	✓	Episode	Trajectory cost	Uniform with the elite samples
PI2	✗	Time-step	Cost to go	Normalized exponentiation
PI2-CMA	✓	Time-step	Cost to go	Normalized exponentiation
PIBB	✗	Episode	Trajectory cost	Normalized exponentiation
PIBB-CMA	✓	Episode	Trajectory cost	Normalized exponentiation
COACH+PS	✓	Time-step/episode	Trajectory cost	Normalized exponentiation

work was proposed for training policies parameterized with linear models of basis functions, and it has been successfully applied to problems like the well known cart-pole and the ball-dribbling with biped humanoid robots. The learning process is supported in the following modules.

Policy supervised learner

The policy $P(s)$ is a linear combination of the feature vector $f \in F$ and the weights vector θ ($P(s) = f^T\theta$). When a human teacher advises a correction, he/she is trying to modify the executed $P(s)$ in the direction of the advice h; therefore, this module updates the weights of the policy model (θ), using a SGD based strategy. But for this model, the error prediction is unknown, since the teacher provides the correction trend, not the exact value of the action to be executed, which could not be intuitive to the human teachers. COACH has the error assumption that sets the prediction error as:

$$error = h \cdot e \tag{7}$$

with h and e as sign and magnitude of the error, h is the teacher feedback, and e is a constant value defined by the user before the learning process. In order to apply the teacher correction, the squared error is used as cost function $J(\theta)$ to be minimized with SGD.

$$\theta = \theta - \alpha \nabla_\theta J(\theta) \tag{8}$$

Therefore, incorporating (7) into (8) and considering the policy $P(s)$ a linear model of the features f, the updating term is

$$\theta = \theta + \alpha \cdot error \cdot \frac{\partial P(s)}{\partial \theta} = \theta + \alpha \cdot h \cdot e \cdot f \tag{9}$$

wherein α is an adaptive learning rate or step size, calculated from the magnitude of $\alpha(s) = |H(s)|$ in each time step, and $H(s)$ is the prediction of the human correction that is described below.

Human feedback modeling

$H(s)$ is a model learned during the training process, that predicts the human corrective advice with respect to the current state s. It maps $H : S \rightarrow \mathbb{R}^D$ that is constrained in the range $[-1, 1]$ in every dimension. Negative predictions mean that the teacher would advise "to decrease" the action magnitude associated to the corresponding axis, while positive computations correspond to predictions of the advice: "to increase" the action. The prediction is used to compute the adaptive learning rate $\alpha(s)$ used for the policy $P(s)$ updating. The absolute value of the prediction defines how much the policy is modified when the teacher advises a correction to the policy in the state s. For instance, if the prediction is very close to zero, it means that in previous corrections for the same region of S, the advice has been alternated between 1 and -1, so the correction tends to be smaller because the teacher is fine tuning around a specific action magnitude.

Human feedback supervised learner

This module updates the parameters vector ψ of the Human Feedback model $H(s) = f^T\psi$, using SGD like in the $P(s)$ updating. These weights are updated similarly to (7) and (8), but in this case the error is known

$$error_{prediction} = h - H(s). \tag{10}$$

The prediction error is given by the difference between the actual human advice h and the prediction given by $H(s)$. The basis vector f is the same used in the Human feedback Model $H(s)$ and the Policy Model $P(s)$.

Credit Assigner

This module is necessary for problems of high frequency, in which human teachers are not able to independently advise the executed continuous action at each time step. This Credit Assigner module is borrowed from the TAMER framework (Knox and Stone 2009), it tackles this problem by associating the feedback not only to the last state-action pair, but to several past state-action pairs. Each past state-action pair is weighted with the corresponding probability c_t (11) computed from the probability density function pdf_{delay} that characterizes the human response delay.

$$c_t = \int_{t-1}^{t} pdf_{delay}(x)dx \tag{11}$$

In this process a new feature vector is computed by the Credit Assigner (f^{cred}), which is the weighted sum of the feature vectors from the past states; this is the actual vector used by the Human Feedback and the Policy Supervised Learners. This credit assigner can be seen like an eligibility traces module for the human correction and implementation. Details can be found in the original paper. This module might not be necessary in problems of low sampling frequency, e.g., 1 Hz.

Algorithm 3: COACH with linear models of basis functions.

```
1:  $e \leftarrow constant$  // error magnitude of the policy model
2:  $\beta \leftarrow constant$  // learning rate of the human model
3:  for $1 \leq t \leq n$
4:    $c_t \leftarrow assignCredit(t)$
5:  end for
6:  while true do
7:    $s \leftarrow getStateVec()$
8:    $f \leftarrow getFeatures(s)$
9:    $P(s) \leftarrow f^T \cdot \theta$
10:   $TakeAction(P(s))$
11:   wait for next time step
12:   $h \leftarrow gethumanCorrectiveAdvice()$
13:   if $h =! 0$
14:     for $1 \leq t \leq n$
15:       $f^{cred} = f^{cred} + (c_t f)$
16:     end for
17:     $H(s) = f^T \cdot \psi$
18:     $\Delta\psi = \beta \cdot (h - H(s)) \cdot f^{cred}$
        $\psi = \psi + \Delta\psi$
19:     $\alpha(s) = | H(s) | = | {f^{cred}}^T \cdot \psi |$
20:     $error \leftarrow h \cdot e$
21:     $\Delta\theta = \alpha(s) \cdot error \cdot f^{cred}$
        $\theta = \theta + \Delta\theta$
22:   end if
23: end while
```

The complete COACH framework is illustrated in Fig. 1, and described in Algorithm 3, where at the beginning the credit assigner module computes the probabilities (11) for weighting the n previous time steps relative to the current state (line 4). These weights are used in the line 15 for computing the feature vector. The amount of n time steps considered in the credit assigner window depends on the sampling frequency of the task, and the probability density function used for modeling the human delay (see Knox and Stone 2009). Every time step the environment is observed and the policy executed (lines 7–11). Time steps wherein the human teacher provides feedback, the human model and the policy are updated taking into account the mentioned assumptions (lines 14–21).

In this work COACH is combined with PS in order to have human guidance based on corrective advice for the PS progress.

3 Policy search guided with human corrective feedback

Policy Search RL algorithms, as any autonomous RL algorithm, take only the feedback given by the cost or reward function and along with the exploration strategy, the search of the solution is performed. This work proposes to combine episode-based PS with human guidance, where the human teacher is able to correct the policy every time step, whereas the PS only updates the policy model after each iteration of M trials, based on the global performance measurement of every roll-out. This combination of feedback sources is presented in two different approaches, where the learning agent is updated by the users with the COACH framework and the PS algorithm in either a sequential or a simultaneous fashion as explained below.

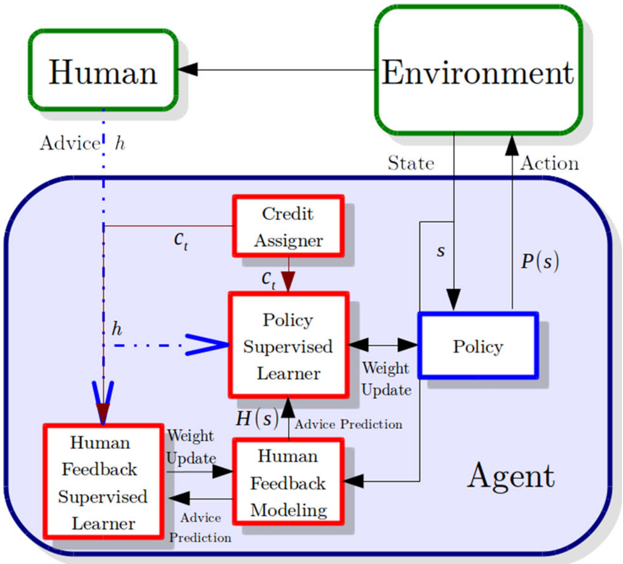

Fig. 1 Architecture of COACH

3.1 Learning sequentially with COACH and policy search

In this scheme called Sequential COACH+PS, we propose to have two independent learning phases, as depicted in Fig. 2. At the beginning, the human teacher interacts with COACH providing corrective feedback to the agent during the task execution. The learning could be from scratch or the initial policy parameters can be set from a policy that needs to be refined. The interactive learning process is executed during several episodes of learning until the user considers that he/she cannot improve the quality of the policy further. Thereafter, the resulting policy is taken as the initial set of parameters for the normal PS process. It has the objective of performing a refinement of the learned parameters in order to achieve an optimal point while locally exploiting the parameters space. The learning process finishes with the same criteria of convergence used when learning only with PS.

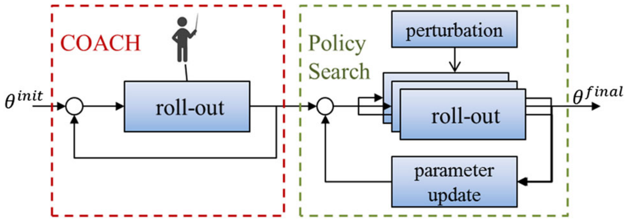

Fig. 2 Learning sequentially with COACH+PS

3.2 Learning simultaneously with COACH and policy search

This second approach named Simultaneous COACH+PS uses simultaneously both sources of information, human feedback and parameter update of PS, for training an agent as shown in Fig. 3. Since the learning progress with COACH is completely based on the human teacher criteria, the convergence is prone to be unstable when the users provide mistaken feedback. Sequential COACH+PS also suffers from this sensitivity to human errors in its first stage of human advice.

We propose to have the PS algorithm at a supervisory level of COACH, in which the cost function determines whether the policy trained by the human teacher is evolving properly or not. Previously we stated that this work is based on episode-based PS, however the proposed simultaneous PS and COACH learning strategy can be seen as a PS algorithm in which the evaluation is episodic, but the exploration is step-based and completely given by the corrections of the human teacher, wherein the changes that COACH obtains over the policy vector are taken as exploration noise by the PS algorithm.

For this hybrid learning scheme, during each roll-out, a regular COACH process of interactive training is carried out (Algorithm 3). From the PS point of view, the evaluation and update stages are kept exactly the same as defined by episode-based PS in Algorithm 2 at lines 3 and 4 with Eqs. (2)–(5), depending on the used algorithm.

The exploration stage is significantly different from the strategy given by (1), because the samples are only created from the distribution $\mathcal{N}(\theta, \Sigma)$ when it is detected that the user is not providing corrective advice anymore. However, during roll-outs advised via COACH, the users advise the agent incrementally, so little by little the policy is improved with the knowledge the users have about the previous policy. Once a teacher started providing feedback, going back to creating independent samples from the current policy distribution for every roll-out tends to confuse the user. In this scenario, some of the obtained samples can result in behaviors completely contrary to what has been taught by the user in the immediately previously executed roll-out. They can interpret this situation as “the agent is not interacting properly or is rejecting the results of the advice after each episode”. This situation diminishes the user interaction with the agent. Therefore, the system would not benefit from the human knowledge.

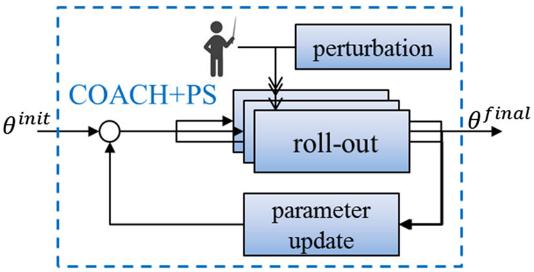

Fig. 3 Learning simultaneously with COACH+PS

To avoid this problem, in this hybrid learning strategy, the resulting policy of a roll-out is set as the initial policy for the next one. There are two different parameter vectors for each roll-out, the vector $\theta^{[m]}$ which is the initial parameter set at the m-th roll-out advised with COACH, and the vector $\theta'^{[m]}$ that is the parameters set returned at the end of the roll-out. The difference of these two vectors is associated to the corrections advised by the teacher.

This sequential exploration allows the teachers to have insights of the operation of the policy, and notions of how to keep improving it through the next roll-out executions. After M trials, the PS updates the policy. In cases where the teacher is continuously improving the task execution, the parameters computed for θ_{k+1} have to be similar to the ones of the most recent task executions. According to the cost function, when the human feedback is harming the policy, the update stage should result in a set of parameters similar to the ones that controlled the agent, before the erroneous human feedback.

Algorithm 4: Policy Search with Simultaneous Human Guidance.

```
1:  repeat
2:    Explore: first roll-out with the current policy $\pi_k$: $\theta^{[1]} = \theta_k$
3:      for $m = 1 \dots M$
4:        $[\theta'^{[m]}] \leftarrow RunRollOut_{COACH}(\theta^{[m]})$
5:        if $\theta'^{[m]} \neq \theta^{[m]}$
6:          $\theta^{[m+1]} = \theta'^{[m]}$
7:        else
8           $\theta^{[m+1]} \sim N(\theta_k, \Sigma_k)$
9:        end
10:       $\theta^{[m]} = \theta'^{[m]}$
11:     end
12:   Evaluate: cost of each roll-out $R^{[m]} = \sum_{t=0}^{T} r_t^{[m]}$
13:   Update: compute new policy parameters using
              $\{\theta^{[m]}, R^{[m]}\}$
14: until Policy converges $\pi_{k+1} \approx \pi_k$
```

Algorithm 4 shows that for every k-th iteration, the first roll-out is executed with the actual parameters vector of the current policy (line 2). The execution of every roll-out is given by $RunRollOut_{COACH}(\theta^{[m]})$ that has the policy vector as input argument, and returns the corrected policy (line

4). When the vector returned at the end of the trial is identical to the initial one, the teacher did not provide any advice and, consequently, the exploration can be carried out by the PS algorithm with the regular strategy (line 8). Regardless whether the roll-out includes teacher advice or not, the policy vector $\theta'^{[m]}$ returned after each execution is added to the set of samples (line 10) for the update stage.

4 Experiments and results

The proposed hybrid learning approaches are tested using simulated and real problems. The proposed sequential and simultaneous approaches are validated and compared to standard Policy Search methods. First, we present an experiment where a previously trained policy is used to simulate a human teacher that interacts with the learning methods in order to train an agent to perform a task. This experiment with simulated teachers is carried out for evaluating the convergence of the learning systems in a setup with controlled human factors, i.e., the experiment only evaluates the capacity of the algorithms to track the intention of the teacher for approximating the policy.

Second, validation experiments with actual human teachers are carried out for learning balancing tasks in simulated and real environments. Finally, we compare the performance of users tele-operating the systems with the controllers learned interactively. This third comparative analysis is intended to show the performances that non-expert users can reach when they are teachers.

For all the experiments of this section the meta-parameters for the PS operation along with the ones related to COACH are fixed. The amount of roll-outs for each PS iteration was set to $M = 10$; in the case of CEM the amount of elite samples is $M_e = 5$. For COACH, the learning rate for the Human Feedback modeling was set $\beta = 0.3$, whereas the error magnitude e was fine tuned for every problem with an initial magnitude that is half of the action range. Any change of this magnitude has proportional impact on the action updating, i.e., it is intuitive to tune for the user.

4.1 Learning with a simulated human teacher

A first evaluation of the proposed methods is presented in this section, wherein a pre-trained policy is used for emulating a human teacher that advises the learning agent. The need to implement non-real human teachers is motivated by two reasons: (i) To evaluate robustness of the learning methods to mistaken human corrections. Unfortunately it is not easy to quantify percentage of mistakes with real human teachers. (ii) To evaluate the learning strategies for correcting the policies themselves in a transparent setup without influence of human factors. Humans not only perform mistakes, but also their attention and effort vary according to their motivation, mood and other factors (e.g., a unmotivated user who never advises corrections, never does mistakes, but also does not transfer any knowledge).

This experiment is carried out using the well known "cart-pole" RL problem (Sutton and Barto 1998), and our implementation is based on this environment.[1] The objective of the task is to learn the forces applied to a cart in order to keep the attached pole balanced. The observed states are the position and velocity of the cart along with the angle of the pole and its velocity respectively $s = [p, \dot{p}, \vartheta, \dot{\vartheta}]$. Usually the goal is to keep the pole balanced, but in this work the complexity of the task is slightly increased by the cost function to minimize $C(T) = \sum_{t=0}^{T} |p_t| - 1$, which requires the pole to be balanced with the cart in the center of the environment in $p = 0$. The episodes start with the cart placed in the center and the pole balanced in upright position; the episode finishes if the pole falls over, the cart reaches the boundaries of the environment, or after 5000 time steps. The policy is parameterized by 4 radial basis functions (RBF) per state variable, for a total of 256 RBF features in the vector $f(s)$. For these experiments, 50 runs of each algorithm with constant parameters were carried out. The figures show the average learning curves and their standard deviations.

4.1.1 Policy search comparison

First, we compare the PS algorithms for learning to solve the cart-pole problem. The task is approached with agents performing the CEM, PI^{BB}, and PI^{BB}-CMA strategies. Each algorithm was run 50 times, and the statistical results are presented in Fig. 4. The learning curves show that PI^{BB}-CMA agents obtain better final performances than the other two PS approaches. Both PI^{BB}-like agents have slightly faster convergence than CEM, and reach considerably lower costs. Additionally, the covariance adaptation component of PI^{BB}-CMA improved the performance by 10% compared to the original PI^{BB}. Therefore, this faster algorithm is used in the experiments of the hybrid approaches proposed in this work.

4.1.2 Hybrid agents with simulated teacher comparison

The emulated human teacher is a block added to the learning loop, that contains the pre-trained policy $P_{pt}(s)$ from Sect. 4.1.1. The objective of this block is to advise corrections like humans do to the learning policy $P(s)$, in order to converge to a similar performance of $P_{pt}(s)$. The corrections h are the same kind of vague binary signals that users provide to a COACH agent, e.g., increase or decrease the executed action. Since the human teachers do not provide cor-

[1] https://github.com/david78k/pendulum/tree/master\penalty\z@/matlab/SARSACartPole.

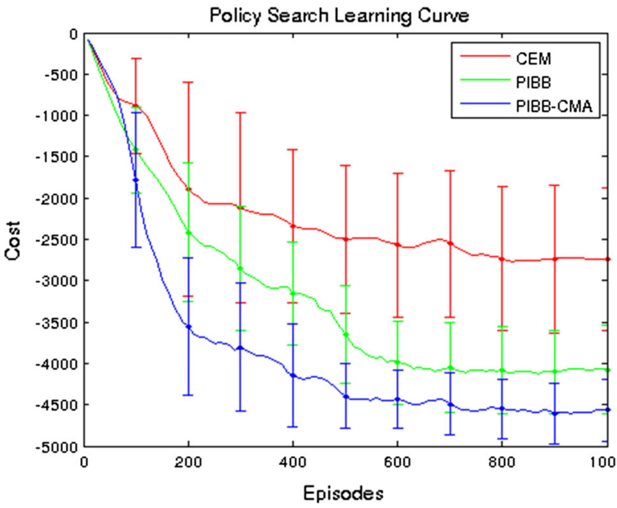

Fig. 4 Learning curves of PS agents with the cart-pole task

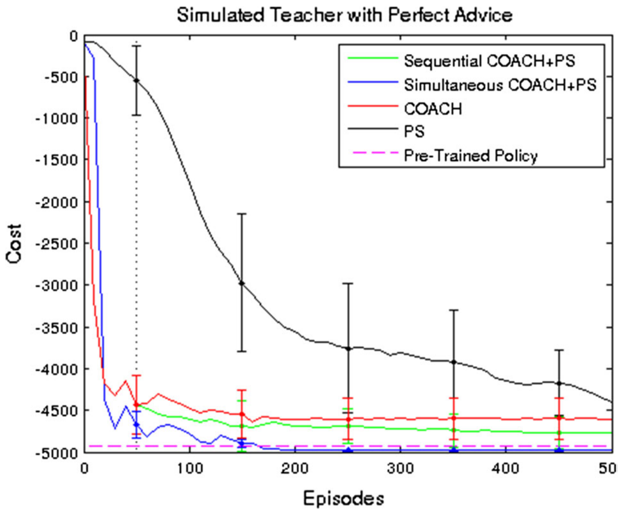

Fig. 5 Learning curves of hybrid agents with simulated human teachers without mistakes

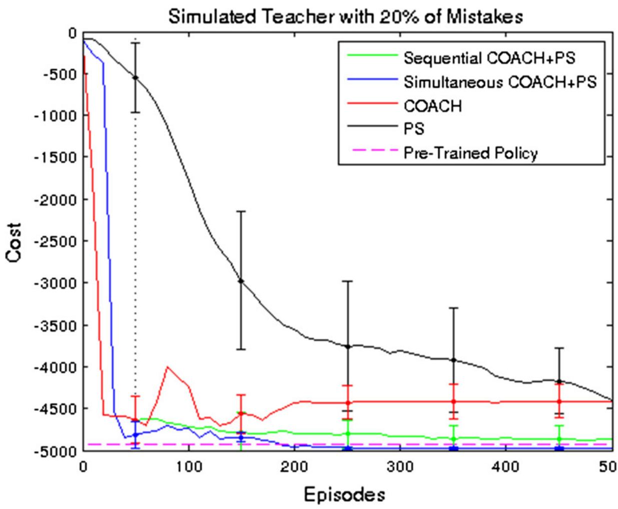

Fig. 6 Learning curves of hybrid agents with simulated human teachers with 20% of mistaken advice

rections in every time step the policy is executed, in this block the frequency of the advice is controlled by a probability ϵ. Therefore, the correction is computed every time step as:

$$h = \begin{cases} \operatorname{sign}(P_{pt}(s) - P(s)) & \text{with probability } \epsilon \\ 0 & \text{with probability } 1-\epsilon \end{cases}, \tag{12}$$

where h would be the “human corrective advice” of line 12 in Algorithm 3. The probability ϵ is diminished after every roll-out with a decay factor of 0.95.

In these experiments, we also consider the fact that in general mistakes are always inherent to human feedback or demonstrations. Therefore, for the simulated teacher, wrong corrective advice is provided to the agent with a probability η as

$$h = \begin{cases} -h & \text{with probability } \eta \\ h & \text{with probability } 1-\eta \end{cases}. \tag{13}$$

The probability η was varied for running learning processes with 0, 20, and 40% of mistakes. In the case of the Sequential COACH+PS, the human feedback is given to the COACH agent only during the first 50 roll-outs, this is indicated with a dotted line in the plots. Then, the PS exploration continues the learning process. Additionally, the hybrid approaches are compared to the performance of the pure COACH agent, the PI^{BB}-CMA approach previously presented in Fig. 4, and the performance of the pre-trained agent used for simulating the human teacher using (12).

The experiments show that for this task, approaches that use the “human feedback” learn faster than the pure Policy Search agent. COACH-only agents have the highest improvement during the very first episodes, however they converge to lower performances than the hybrid agents. Moreover, the performance of COACH is the one most sensible to mistaken feedback, since its convergence decreases more than the other agents when the probability of mistakes η is higher. The previous observation is expected because the human advice is the unique source of information for COACH. Hence, the policy performance only depends on the quality of the feedback. This is one of the motivations for combining COACH with RL.

Sequential COACH+PS shows the most stable convergence, since from the 50th episode on, there is no human advice that might harm the already good policies obtained with COACH. The results of the experiments with 0 and 20% of mistakes (Figs. 5, 6), show that in the moment when the stage of learning with COACH stops and the PS phase starts, the learning curve turns into the least steep but the

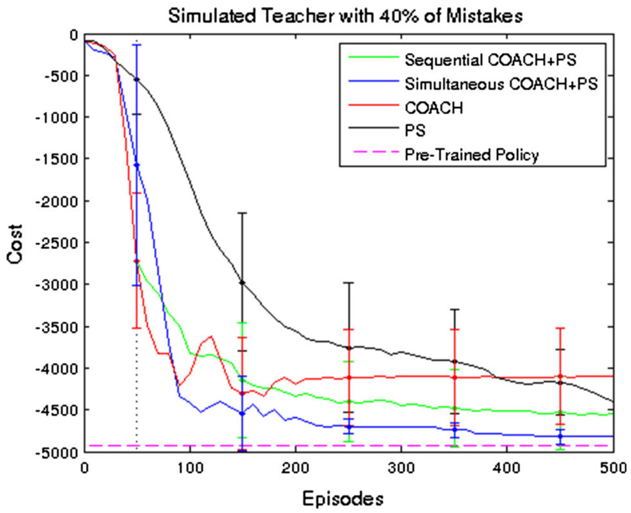

Fig. 7 Learning curves of hybrid agents with simulated human teachers with 40% of mistaken advice

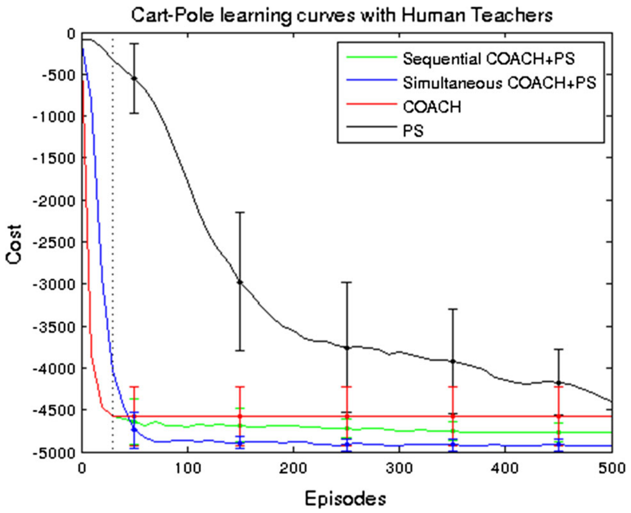

Fig. 8 Learning curves of the experiments for the cart-pole problem with real human teachers

most monotonic. This happens as PS can be seen as a fine tuning of the policy obtained by the COACH stage, which quickly learns good policies. In the case of 40% of mistakes in Fig. 7, the PS stage drastically improves the performance, the learning curve looks like the pure PS convergence, but displaced to the left by 300 trials.

The Simultaneous COACH+PS approach obtains the best final performances of the policies compared to the other agents, it even outperforms the pre-trained policy used for simulating the teacher when there is 0 and 20% of erroneous feedback. The simultaneous combination of interactive learning and PS makes the learning slower in the very first episodes compared to COACH. However, it outperforms COACH's performances after approximately 30, 40, and 90 episodes in the cases of "human feedback" with 0, 20, and 40% of incorrect advice respectively.

These experiments show that hybrid approaches are more robust to noisy corrections, and faster than pure autonomous or pure interactive learning approaches.

4.2 Learning with real human feedback

A more detailed validation of the proposed methods is carried out with experiments involving real human teachers interacting with simulated and real problems, in which the corrective feedback is provided to the agent with a keyboard. Again, the cart-pole problem is approached along with a real inverted pendulum swing-up (Adam et al. 2012). In these experiments five participants interacted with the agents as teachers. They advised Simultaneous COACH+PS agents during the episodes they considered appropriate. The users also interacted with the COACH agent during 30 episodes, and the obtained policies were used for the initial policy in the second stage of the Sequential COACH+PS. The learning curves are also compared to those obtained by pure PS agents. The video[2] shows the interactive learning and execution of the agents.

4.2.1 Cart-Pole

In this validation, the same environment used in the previous experiments is used for learning to execute the task with support of real humans. Results in Fig. 8 show that all the agents that use human advice obtain five times faster convergence than pure PS, and that the hybrid agents obtain better policies than pure COACH or pure PS. The Simultaneous COACH+PS obtains the best performances, but in the very first episodes is slower than COACH, and consequently slower than the sequential scheme. For this problem it is possible to see that the convergence is more monotonic than the experiments with simulated human teachers, although lower final performances are obtained. This can be due to the capacity of real human teachers that are adapting to advise the current policy, leading to different final policies with similar performance, whereas, the simulated teachers only try to teach to imitate the pre-trained policy.

4.2.2 Pendulum swing-up

The second experiment is carried out using an under-actuated inverted pendulum swing up, which is a weight attached to a DC motor depicted in Fig. 9. The observed states are the angle and its velocity $s = [\vartheta, \dot{\vartheta}]$, while the action is the voltage u applied to the motor, which is in the range -2 to 2V. As the motor does not have enough force to rotate the pendulum up directly, the first problem to solve is to learn to swing the

[2] https://youtu.be/VIJiK7Rhe4o.

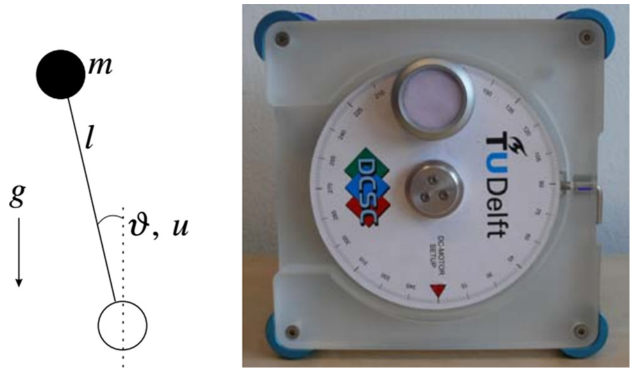

Fig. 9 Inverted pendulum swing-up setkup (taken from Adam et al. 2012)

weight back and forth for reaching the upper position ($\vartheta = \pi$). Then, the second problem is to learn to keep the pendulum balanced in the unstable equilibrium. In this case, each state variable is split into 20 RBF features (20 × 20) for a total of 400 features that compound the vector f. The episodes are terminated after 500 time steps (10s), and the cost function is given by $C(T) = \sum_{t=0}^{T} -(\vartheta_t/\pi)^4$, which is engineered to be sensitive to small angles close to the upright position.

For this problem COACH uses an Inverse Kinematics (IK) model to map the advice from the effector space to the actuator space. The user provides corrections like “move the pendulum more towards right, left, up, or down” with the arrows of a keyboard, and this module based on the current state, uses the IK for mapping the advice into “apply more or less voltage to the motor”.

The results obtained with the real system are very similar to the simulations (both in Fig. 10). In this new set of experiments we again observe that all interactive approaches are faster than the pure PS. However, the performances obtained with only COACH are outperformed by PS after several episodes. The PS agent takes seven times more episodes to reach the performance reached by the users after 20 trials advised with COACH. The hybrid schemes have the best cost indices; the sequential approach successfully employs PS to fine-tune the initial policy obtained via COACH. As before, the Simultaneous COACH+PS has slower convergence than pure COACH in the first episodes, but keeps improving until reaching the highest performances.

Simultaneous COACH+PS is slower than Sequential COACH+PS at the beginning, because in the simultaneous scheme, the probability weighted average computed by the PS can be compared to a lowpass filter, that avoids drastic changes in the parameters that might obtain considerable positive or negative impact on the performance. Nevertheless, during the episodes before convergence, the occasional human feedback (not present in Sequential COACH+PS at that moment) seems to be more efficient than the random exploration given by the normal distribution of (1).

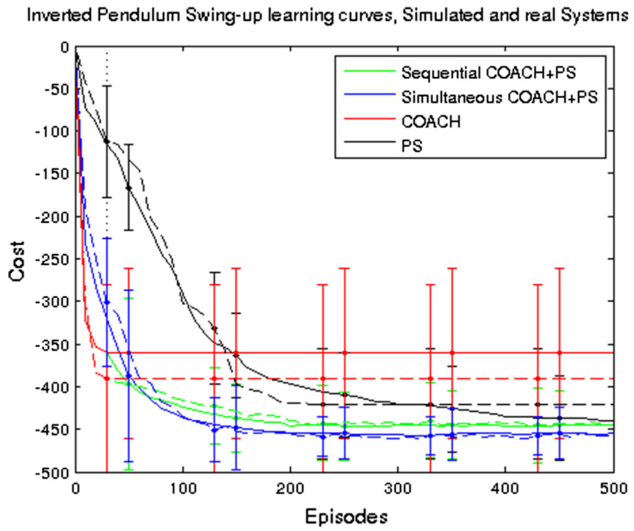

Fig. 10 Learning curves of the experiments for the inverted pendulum swing-up problem with the simulated system (normal lines) and the real system (dashed)

4.2.3 3DoF arm inverse kinematics

This third evaluated problem is about learning the inverse kinematics model for a real 3 DoF robot arm (Fig. 11). The model has to map the input request of a 3D coordinate position into the space of the angles of the three servos that compose the robot arm, which is the output. The cost function used in this problem is the Euclidean distance between the points requested to the model and the actual arm’s end effector position, which is given in centimeters.

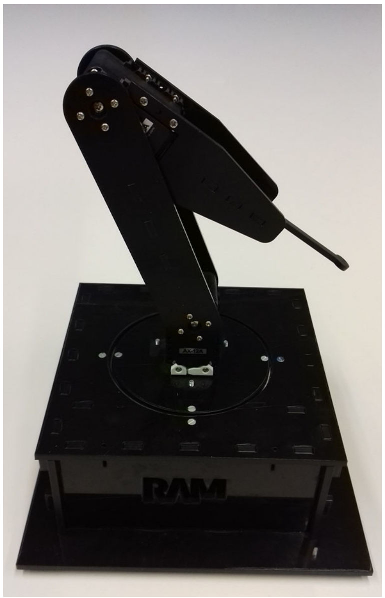

Fig. 11 Robot arm used for learning the IK model. The robot was designed and built by the Robotics and Mechatronics group, University of Twente, the Netherlands

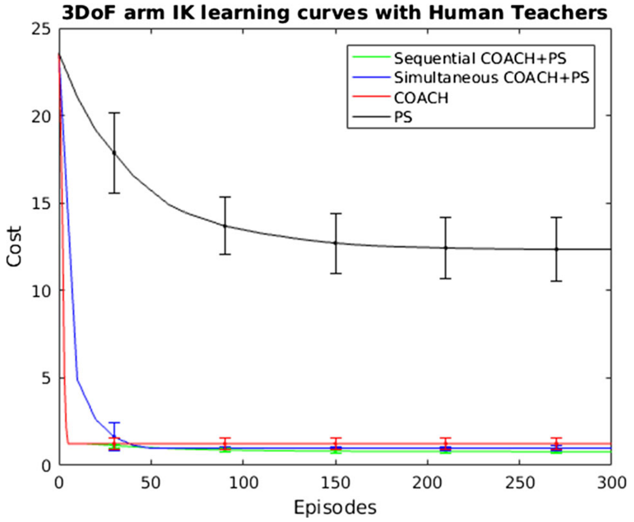

Fig. 12 Learning curves of the experiments for the inverse kinematics for a 3 DoF robot arm. Average Cost in cm

In this application, when the human teachers interact with the learning robot, they observe the position of the end effector with respect to the target point, then they provide corrective advice in the joints space for decreasing the distance. This robot does not have an operation mode for kinesthetic teaching, so this corrective advice is the only way to obtain human feedback in the action domain. For the experiments of learning with COACH and Sequential COACH+PS, the human teachers only interact with the robot during the first five episodes, whereas with Simultaneous COACH+PS the users continue advising when considered necessary.

The results obtained from the learning processes in Fig. 12 show similar trends as the previous experiments. In this case, the PS algorithm, which is a local search methods, converges to local minima. With the interactive agents, the obtained costs are considerably lower than the ones reached with only PS. Again Simultaneous COACH+PS showed to be slowest at the first episodes, however it outperforms the performance of pure COACH after 40 roll-outs. For this problem the sequential hybrid algorithm converge to a slightly lower cost than the simultaneous counterpart, however both reduce 40% of the error obtained with only COACH, thus the average error of the hybrid agents is 7 mm, while in the best run the policy model converged to an error of 3 mm.

In this experiment, the human knowledge leveraged the convergence of the policy, the visual perception of the human teacher can help in the very first part of the learning process, nevertheless, that can bring on inaccuracies, especially for sensing depth. However, the RL component of the algorithms, that evaluates based on the cost function supports the refinement of the model for attaining a better accuracy.

4.3 Comparison between tele-operation and learning

In this set of experiments, the abilities of the users for teaching the agents to perform the tasks are compared to the capacities of the users for actually executing the tasks with tele-operation. The participants interacted with the system several trials for learning to execute the tasks. Their best execution is compared to the performance obtained using COACH. Additionally, an inverted wedge is employed as a fourth case study; for this balancing task the cost function is the average angle per episode, where the equilibrium point is zero degrees. This task is illustrated in the video.

Table 2 shows the results comparing the users tele-operation and teaching; since the cost functions have to be minimized, the lower the index the better the performance. For all the cases explored in this work, the users did not learn to tele-operate the systems successfully, as reflected in Table 2. They kept the equilibrium only for a few seconds for all the problems except for the pendulum swing up, in which the pendulum was never balanced. It is interesting to observe that the users can obtain agents which can perform tasks that the human teachers cannot demonstrate, i.e., the interactive learning approaches based on advise of corrections let non-expert users in the task domain teach policies of good quality from vague binary pieces of advice. The numeric results in Table 2 show the big difference between the two options of interaction of users with the systems (operating and teaching). This highlights the advantages of sharing the work with

Table 2 Comparison of users tele-operating and teaching

Task	Performances	
	Tele-operation	Interactive learning
Cart-pole	– 249.13	– 4576.43
Simulated pendulum swing-up	– 81.79	– 360.52
Real pendulum swing-up	– 68.44	– 390.55
Simulated inverted wedge	0.2276	0.07024
Real inverted wedge	0.2718	0.0838

intelligent systems that can learn from the users who are not able to provide good demonstrations. These particular features of the hybrid approaches are desirable in environments where users frequently need to adapt the agent to new conditions or tasks.

5 Conclusion

This work has proposed and validated approaches for policy search supported with human feedback. Two schemes of combining PS with an interactive framework based on corrective instructions were presented: a sequential scheme and a scheme that learns simultaneously from human and autonomous feedback.

The experiments with balancing tasks and the inverse kinematics model showed that the hybrid algorithms can benefit from the advantages of both kinds of learning strategies, where the corrections provided by human teachers result in fast learning to a high but suboptimal performance, whereas PS can optimize policies based on cost functions that are not very explicit or intuitive to the users' understanding, or simply when the human perception becomes too limited to support the learning process. Therefore, the addition of human support to PS speeds up the convergence between 3 to 30 times according to the results obtained. From the point of view of interactive machine learning, these hybrid strategies provide more robustness to the convergence, since the sensitivity to noisy or mistaken corrections is diminished. Moreover the quality of the policies is improved with the cost based corrections of PS which perform fine tuning of the policies taught by the users.

The proposed hybrid schemes showed to be better choices than pure COACH or PS frameworks in applications that need fast learning. Sequential COACH+PS is a simple scheme easy to implement and completely agnostic of the type of PS used; it facilitates the learning especially in the first trials. Simultaneous COACH+PS scheme showed to be slower than the sequential one at the very beginning. However, it benefits from the occasional corrections given by the teachers, which guide the exploration to the highest achieved performances.

The comparison of all the learning approaches, and even the performance of the users tele-operating the agents, shows that the proposed strategies can have high impact on cyber-physical systems of the coming industrial developments, that require to reduce the workload of factory operators. and also to ease the adaptability of the products for the final users, who can interact for modifying the operation of technological products.

Acknowledgements This work was partially funded by FONDECYT project 1161500 and CONICYTPCHA/Doctorado Nacional/2015-21151488

References

Abbeel, P.& Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In: *Proceedings of the twenty-first international conference on Machine learning*, p. 1. ACM

Adam, S., Busoniu, L., & Babuska, R. (2012). Experience replay for real-time reinforcement learning contro. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews*, *42*(2), 201–212.

Akrour, R., Schoenauer, M., & Sebag, M. (2011). Preference-based policy learning. *Machine learning and knowledge discovery in databases* pp. 12–27

Akrour, R., Schoenauer, M., Sebag, M., & Souplet, J. C. (2014). Programming by feedback. In: *International Conference on Machine Learning*, 32, pp. 1503–1511. JMLR. org

Argall, B. D., Browning, B., & Veloso, M. (2008). Learning robot motion control with demonstration and advice-operators. In: *International conference on intelligent robots and systems, 2008. IROS 2008. IEEE/RSJ*, pp. 399–404.

Argall, B. D., Browning, B., & Veloso, M. M. (2011). Teacher feedback to scaffold and refine demonstrated motion primitives on a mobile robot. *Robotics and Autonomous Systems*, *59*(3), 243–255.

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

Atkeson, C. G., & Schaal, S. (1997). Robot learning from demonstration. In: *ICML*, Vol. 97, pp. 12–20.

Busoniu, L., Ernst, D., De Schutter, B., & Babuska, R. (2011). Cross-entropy optimization of control policies with adaptive basis functions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *41*(1), 196–209.

Celemin, C., & Ruiz-del Solar, J. (2015). Interactive learning of continuous actions from corrective advice communicated by humans. In: *Robot soccer world cup*, pp. 16–27. Springer

Celemin, C., & Ruiz-del Solar, J. (2018). An interactive framework for learning continuous actions policies based on corrective feedback. *Journal of Intelligent and Robotic Systems*. https://doi.org/10.1007/s10846-018-0839-z.

Chernova, S., & Thomaz, A. L. (2014). Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *8*(3), 1–121.

Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., & Amodei, D. (2017) Deep reinforcement learning from human preferences. arXiv:1706.03741

Deisenroth, M. P., Neumann, G., Peters, J., et al. (2013). A survey on policy search for robotics. Foundations and Trends®. *Robotics*, *2*(1–2), 1–142.

Heidrich-Meisner, V., & Igel, C. (2008). Evolution strategies for direct policy search. In: *PPSN*, pp. 428–437. Springer

Jain, A., Wojcik, B., Joachims, T., & Saxena, A. (2013). Learning trajectory preferences for manipulators via iterative improvement. In: *Advances in neural information processing systems*, pp. 575–583

Knox, W. B., & Stone, P. (2009). Interactively shaping agents via human reinforcement: The tamer framework. In: *Proceedings of the fifth international conference on Knowledge capture*, pp. 9–16. ACM

Knox, W. B., & Stone, P. (2012). Reinforcement learning from simultaneous human and mdp reward. In: *Proceedings of the 11th* *international conference on autonomous agents and multiagent systems* Vol. 1, pp. 475–482.

Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, *32*(11), 1238–1274.

Kober, J., Wilhelm, A., Oztop, E., & Peters, J. (2012). Reinforcement learning to adjust parametrized motor primitives to new situations. *Autonomous Robots*, *33*(4), 361–379.

Levine, S., Finn, C., Darrell, T., & Abbeel, P. (2015) End-to-end training of deep visuomotor policies. arXiv:1504.00702

Levine, S., Pastor, P., Krizhevsky, A., & Quillen, D. (2016). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. arXiv:1603.02199

Mannor, S., Rubinstein, R. Y., & Gat, Y. (2003). The cross entropy method for fast policy search. In: *ICML*, pp. 512–519

Najar, A., Sigaud, O., & Chetouani, M. (2016). Training a robot with evaluative feedback and unlabeled guidance signals. In: *25th IEEE international symposium on robot and human interactive communication (RO-MAN), 2016*, pp. 261–266.

Ng, A. Y., & Russell, S. J., et al. (2000). Algorithms for inverse reinforcement learning. In: *ICML*, pp. 663–670

Pilarski, P.M., Dawson, M.R., Degris, T., Fahimi, F., Carey, J.P., & Sutton, R.S. (2011). Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning. In: *IEEE international conference on rehabilitation robotics (ICORR), 2011* , pp. 1–7.

Rozo, L., Jiménez, P., & Torras, C. (2013). A robot learning from demonstration framework to perform force-based manipulation tasks. *Intelligent Service Robotics*, *6*(1), 33–51.

Stulp, F., & Sigaud, O. (2012a). Path integral policy improvement with covariance matrix adaptation. arXiv:1206.4621

Stulp, F., & Sigaud, O. (2012b). Policy improvement methods: Between black-box optimization and episodic reinforcement learning

Stulp, F., & Sigaud, O. (2013). Robot skill learning: From reinforcement learning to evolution strategies. *Paladyn, Journal of Behavioral Robotics*, *4*(1), 49–61.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction* (Vol. 1). Cambridge: MIT Press.

Tenorio-Gonzalez, A. C., Morales, E. F., & Villaseñor-Pineda, L. (2010). Dynamic reward shaping: training a robot by voice. In: *Ibero-American conference on artificial intelligence*, pp. 483–492. Springer

Theodorou, E., Buchli, J., & Schaal, S. (2010). A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, *11*(Nov), 3137–3181.

Thomaz, A. L., Breazeal, C., et al. (2006). Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. *Aaai*, *6*, 1000–1005.

Zhifei, S., & Joo, E. M. (2012). A review of inverse reinforcement learning theory and recent advances. In: *IEEE congress on evolutionary computation (CEC), 2012*, pp. 1–8.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Carlos Celemin is a Ph.D. candidate at University of Chile in the Department of Electrical Engineering, member of the Advanced Mining Technology Center, and the UChile Robotics Team that yearly participates in the RoboCup competition. He received the best paper award in the International RoboCup Symposium 2015. His research interests are in machine learning, robotics, human-machine interaction, and decision making systems. He has been a visiting research student in the CORAL Lab at Carnegie Mellon University, the Delft Center for Systems and Control (DCSC) at Delft University of Technology, and the Intelligent Autonomous Systems Group (IAS) at Technical University of Darmstadt.

Javier Ruiz-del-Solar received his diploma in Electrical Engineering and the MS degree in Electronic Engineering from the Technical University Federico Santa Maria (Valparaíso, Chl) in 1991 and 1992, respectively, and the Doctor-Engineer degree from the Technical University of Berlin (Ger) in 1997. In 1998 he joined the Electrical Engineering Department of the Universidad de Chile (Santiago, Chl) as Assistant Professor. In 2001 he became Director of the Robotics Laboratory, in 2005 Associate Professor and in 2011 Full Professor. His research interests include mobile robotics, human-robot interaction, and face analysis. Dr. Ruiz-del-Solar is recipient of the IEEE RAB Achievement Award 2003, RoboCup Engineering Challenge Award 2004, RoboCup @Home Innovation Award 2007, and RoboCup@Home Innovation Award 2008. Since 2006 he has been a Senior Member of the IEEE, and since 2008 a Distinguished Lecturer of the IEEE Robotics and Automation Society. He is currently Director of the Advanced Mining Technology Center at the Universidad de Chile.

Jens Kober is an assistant professor at the Cognitive Robotics department, TU Delft, The Netherlands. He worked as a postdoctoral scholar jointly at the CoR-Lab, Bielefeld University, Germany and at the Honda Research Institute Europe, Germany. He received his Ph.D. in 2012 from Technische Universität Darmstadt, Germany. From 2007 to 2012 he was working at the Department Schölkopf, MPI for Intelligent Systems, Germany. He has been a visiting research student at the Advanced Telecommunication Research (ATR) Center, Japan and an intern at Disney Research Pittsburgh, USA.