Abstract

Learning from human feedback is a viable alternative to control design that does not require modelling or control expertise. Particularly, learning from corrective advice garners advantages over evaluative feedback as it is a more intuitive and scalable format. The current state-of-the-art in this field, COACH, has proven to be an effective approach for confined problems. However, it parameterizes the policy with Radial Basis Function networks, which require meticulous feature space engineering for higher order systems. We introduce Gaussian Process Coach (GPC), where feature space engineering is avoided by employing Gaussian Processes. In addition, we use the available policy uncertainty to 1) inquire feedback samples of maximal utility and 2) to adapt the learning rate to the teacher’s learning phase. We demonstrate that the novel algorithm outperforms the current state-of-the-art in final performance, convergence rate and robustness to erroneous feedback in OpenAI Gym continuous control benchmarks, both for simulated and real human teachers.

1 INTRODUCTION

In contrast to autonomous Machine Learning techniques, humans are very effective in inferring suitable control strategies when facing new problems. Specifically for intuitive problems, like picking up objects or playing simple games, humans are able to achieve decent performance on first try (Hessel et al., 2018). Communicating this domain knowledge has shown to drastically accelerate model-free control techniques. For example, a well known approach is the Learning from Demonstration (LfD) framework, where a policy is derived using examples of proper execution (Ross et al., 2011). Other methods, like Apprenticeship Learning, employ demonstration to reversely derive the trainer’s choices for autonomous improvement (Abbeel and Ng, 2004). To avoid superficial (and possibly expensive) interaction with the trainer, Gräve and Behnke (2013); Losey and O’Malley (2018) improved sample efficiency with an Active Learning (AL) framework where demonstrations are inquired especially for uncertain policy executions.

LfD could however be troublesome for systems that feature agile dynamics. Moreover, the demonstrations require expert knowledge of the system and the solution (Argall et al., 2009). A less demanding approach has been studied by, e.g., Griffith et al. (2013); Knox and Stone (2009), where the trainer gives scalar reward signals (evaluative feedback) in response to the agent’s observed behavior. Thomaz and Breazeal (2006) however argue that trainers implicitly guide in their reward signal, and base their feedback not solely on past actions but also
on what is going to happen. This intrinsic preference in guidance has been studied by Celemin and Ruíz-del Solar (2015) and resulted in COrective Advice Communicated by Humans (COACH), an algorithm that allows teachers to shape the optimal policy by providing corrective feedback, i.e. in the action domain. This approach engages the users intuition without requiring expertise on the task. Moreover, teachers are now able to guide a policy rather than to evaluate it, which has shown to be better scalable to high-dimensional problems (Suay and Chernova, 2011). COACH has shown to be very efficient on intuitive problems and outperforms teachers, to show the significance of the proposed contributions and the performance in a comparison against several benchmarks of the OpenAI gym (Brockman et al., 2016), both with simulated and real human teachers, to show the significance of the proposed contributions and the performance in a comparison against previous work.

This study is organized as follows: background material is covered in Section 2, Section 3 details the novel algorithm GPC. The experimental setup and corresponding results are presented in Section 4 and Section 5 respectively.

2 BACKGROUND

In the following, we detail the key components of GPC, starting with COACH which is the basis of the novel framework. The principles of GP are covered thereafter.

2.1 COACH

COrrective Advice Communicated by Humans (COACH), proposed by Celemin and Ruíz-del Solar (2015), is an algorithm that trains agents with corrective advice. It has policy \( P_c : S \rightarrow \mathbb{R}^n \), with \( S \) the set of states and \( n \) the action-space dimensionality, that maps states to continuous actions. The trainer observes the agent and occasionally suggests to either increase or decrease the action. This feedback \( h \in \{-1, 1\} \) is modelled in the human feedback model: \( H_c : S \rightarrow \mathbb{R}^n \). The parameterization of both models is done by RBF networks, where the respective models have different weight to the feature vector \( \phi(s_k) \), with \( s_k \) denoting the state in time-step \( k \). The learning framework is supported by the following modules.

\textbf{2.1.1 Policy Supervised Learner}

The policy \( P_c(s_k) \) provides the action \( a \) for a given state \( s_k \), by taking the linear combination of the weights and the feature vector, i.e. \( a_k = P_c(s_k) = \theta_k^T \phi(s_k) \), with \( \theta \) the weight vector of the policy. For every directive correction \( h \) given by the teacher, the weight vector is updated according to a Stochastic Gradient Descent approach:

\[ \theta_{k+1} = \theta_k - \alpha(s_k) \nabla J(\theta), \]

with \( \alpha(s_k) \) the learning rate (obtained as described in Section 2.1.2 and \( J(\theta) \) denoting the cost function, which is the squared error between the applied and 'desired' action, given by \( h \) and magnitude \( c \). The latter denotes a free parameter set by the user within the range of the action domain. Hence, taking the human feedback into account, the gradient becomes

\[ \theta_{k+1} = \theta_k + \alpha(s_k) \phi(s_k) he. \]  

\textbf{1: Given:}
- Policy learning rate \( e \)
- Human model learning rate \( e \)
- Constant learning rate \( c \)
- Feature space function \( \phi(\cdot) \)

2: for all \( k \) do
3: Get state \( s_k \)
4: Compute new action \( a_k \leftarrow \theta_k \phi(s_k) \)
5: Obtain corrective advise \( h \)
6: if \( h \neq 0 \) then
7: \( H(s_k) = \psi_k \phi(s_k) \)
8: \( \Delta \psi = \beta(h - H(s_k))\phi(s_k) \)
9: Human model update \( \psi_{k+1} = \psi_k + \Delta \psi \)
10: Get learning rate \( \alpha(s_k) = |H(s_k)| + c \)
11: \( \Delta \theta = \alpha(s_k) \phi(s_k) he \)
12: Policy update \( \theta_{k+1} = \theta_k + \Delta \theta \)
13: end if
14: end for
ear combination of the human model weights $\psi_k$ and the feature vector $\phi(s_k)$, i.e. $H(s_k) = \psi_k \phi(s_k)$. The updates on the weight vector $\psi_k$ are conducted in the same fashion as (1), but now with a known error magnitude $\epsilon_h = h - H(s_k)$ such that

$$
\psi_{k+1} = \psi_k + \beta (h - H(s_k)) \phi(s_k)
$$

with $\beta$ the learning rate of the human model. By this human model, the learning rate in (1) is given as

$$
\alpha(s_k) = |H(s_k)| + c_c.
$$

(2)

Note that $H(s_k) \approx 1$ for consistent feedback with equal sign and hence increases the learning steps. For alternating feedback the learning rate diminishes. To prevent the learning rate from dwindling to zero, (2) is appended with a constant factor $c_c$.

The outline of the COACH framework is depicted in Algorithm 1. Lines 3-5 comprise of policy executions. The model updates (line 7-10) as motivated in Section 2.1.2. The pseudo-code of GPC is in Algorithm 2. Lines 3-5 are subsequently given in lines 11-12. Furthermore, the COACH framework can be extended by the Credit Assigner, which takes a human delay into account for the feedback given by the teacher. Since this study will not exploit this feature it will not be covered here.

### 2.2 GAUSSIAN PROCESSES

Gaussian Processes (GPs) are Bayesian non-parametric function approximation models. It is a collection of random variables, such that every finite collection of those random variables has a multivariate normal distribution. GPs do not require specification of a model structure a priori and provide the uncertainty along with the predictions. A GP is fully specified by its mean $m(x)$ and covariance function $k(x, x')$, i.e.

$$
 f(x) \sim \mathcal{GP}(m(x), k(x, x')).
$$

Let $y = \{y_1, \ldots, y_n\}$ be a set of observations from a stochastic process

$$
 y_i = f(x_i) + \epsilon,
$$

(3)

where $x_i$ denotes the input vector of observation $y_i$. The noise $\epsilon$ is assumed Gaussian with standard deviation $\sigma_\epsilon$. The input matrix is defined as $X = \{x_1, \ldots, x_n\}$. Applying the conditional distributions (Rasmussen and Williams, 2006), the following posterior predictive equations for test inputs $x_*$ are given as:

$$
 f_* | X, y, x_* \sim \mathcal{N} (\tilde{f}_*, \text{cov}(f_*)), \quad \text{where}
$$

$$
 \tilde{f}_* = m(x_*) + K_* [K + \sigma^2_n I]^{-1} (y - m(X)),
$$

$$
 \text{cov}(f_*) = K_* - K_* [K + \sigma^2_n I]^{-1} K_*
$$

where $K_* = k(X, x_*)$, $K_{**} = k(x_*, x_*)$, and $K$ is the Gram matrix with entries $K_{ij} = k(x_i, x_j)$. The Gaussian noise per observation is denoted as $\sigma_n$ and has a similar function as $\epsilon$ in (3). The kernel function $k(x, x')$ is a measure of similarity between two input vectors $x$ and $x'$. In this study, we employ two kernel functions. The first one is the squared exponential (SE) kernel, which is given as

$$
 k_s(x, x') = \sigma^2_s \exp \left( -\frac{\|x - x'|^2}{2\ell^2} \right),
$$

(4)

with $\sigma_s = \{\sigma_s, l\}$ the hyperparameters of the kernel function, consisting of the signal variance $\sigma_s$ and length-scale $l$. The length-scale denotes a measure for the roughness of the data. In general, one can assume that extrapolating more than $l$ units away from the input data is considered unreliable. The second kernel function, the Matérn kernel, is specified as

$$
 k_m(x, x') = \sigma^2_m 2^{1-\nu} \frac{\Gamma(\nu)}{\nu !} \left( \frac{\sqrt{2\nu} |x - x'|}{l} \right) B_v \left( \frac{\sqrt{2\nu} |x - x'|}{l} \right),
$$

(5)

with $B_v(\cdot)$ the modified Bessel function (Abramowitz and Stegun, 1965), $\Gamma(\cdot)$ the Gamma function and the hyperparameters $\beta_m = \{\nu, l, \nu\}$. Here, $\nu$ denotes a ‘smoothness’ parameter that correlates with the amount of times the target function is differentiable (Rasmussen and Williams, 2006).

For multivariate targets, we train conditionally independent GPs for each target dimension.

### 3 GAUSSIAN PROCESS COACH (GPC)

We now introduce Gaussian Process Coach (GPC), an algorithm based on COACH that employs GP as an alternative to RBF networks to comprise advantage in scaling and sample efficiency. A schematic of the method is depicted in Fig. 1. In the main format, the trainer observes the environment and the current policy and provides action corrections to advance the policy. These corrections trigger agent updates in order to take immediate effect on the policy. This process is repeated until convergence. The pseudo-code of GPC is in Algorithm 2.

This section defines GPC for a one dimensional action-space, but scales straightforwardly to higher dimensional problems.

#### 3.1 MODELLING POLICY AND FEEDBACK

The GPC framework engages two GP models: the policy $P$ and the human model $H$. The prior of the policy is modelled as:

$$
 P : S \rightarrow \mathbb{R} \sim \mathcal{GP}(m_p(s), k_p(s, s')),
$$

(6)
The policy in (6) and human model in (7) both concern a multidimensional regression on the input data. Each input dimension may however be subject to data with completely different orders of magnitude, such that a single length-scale is unsuitable. We therefore take an approach that allows us to set an independent length-scale per input dimension.

Let us consider the SE kernel from (4). Following [Rasmussen and Williams, 2006], the parameterization in terms of the hyperparameters results in

$$k_s(x, x') = \sigma_s^2 \exp \left( -\frac{1}{2} (x - x')^T l_s M (x - x') \right),$$

with $M$ the diagonal matrix consisting of the characteristic length-scales per axis. Such a covariance function implements Automatic Relevance Determination (ARD) [Neal, 1995]. This study adopts two distinct methods for determining the diagonal values of $M$. In the first approach we let the trainer decide on the respective relevance of the input dimensions:

$$M_{cs} = \text{diag}(w)^{-2},$$

with $w$ a vector consisting of custom ‘weights’ on the input dimensions. These values are determined a priori and deemed static throughout the learning process. This method is referred to as GPC(-CS). The second method concerns the normalization of the independent inputs for an equal relative dependency, resulting in an approach where any length-scale tuning is circumvented. The result is an approach that does not scale with the input dimension and could therefore be decisive for higher-order systems. The parameterization is carried out by

$$M_{ns} = \text{diag}(\sigma_m)^{-2},$$

with $\sigma_m$ the vector containing the variance of the independent input dimensions, which is updated for every feedback sample (see line 13 in Algorithm 2). This method will be referred to as GPC-NS.

The extension to the Matérn kernel (5) is straightforward with $M_{cs} = \text{diag}(w)$ and $M_{ns} = \text{diag}(\sigma_m)$ for every length-scale $l$. To distinguish between the scaling of the policy and the human model we add subscript $h$ and $p$, e.g. $M_{cs,h}$.

### 3.3 LEVERAGING UNCERTAINTY

GPs provide uncertainty estimates with every query point based on dissimilarity with the training data. For the policy, the uncertainty reflects the presence of feedback data in the respective or surrounding state. Due to the integration of the action in the input of the human model, this...
uncertainty reflects the presence of feedback for state-action pairs. To elaborate on this, a hypothetical example is depicted in Fig. 2. The contiguous plots show the evolution of the policy and its uncertainty as new feedback is obtained. We may envision this principle as building a map that discloses certain and uncertain regions with respect to past feedback. This feature comprises the main advantage of GPC over other methods.

3.3.1 Adaptive Learning Rate (ALR)

We assume that the teacher encounters two teaching phases during the learning period. The initial learning phase arises when the process is commenced and the policy is idle. We believe that the feedback in this stage will mostly concern raw adjustments in order to create a coarse version of the final policy. These coarse adaptations will gradually shift towards the second learning phase where trainers apply small refinements to the policy for meticulous improvements. In this study, we model the transition from coarse to fine adjustments not as a universal annealing process. Instead, we adapt the learning rate to the intended correction per state.

Hence, we introduce the following Adaptive Learning Rate (ALR):

$$r_k = \sigma_p(s_k) + \sigma_h(z_k) + c_r, \quad (8)$$

with $r_k$ the learning rate, $s_k$ the state and $z_k$ the concatenation of $(s_k, a_k)$ (see line 7-9 in Algorithm 2). The uncertainty of the policy $\sigma_p$ allows us to accelerate the learning by increasing the learning rate for the first feedback instances. The uncertainty estimation of $\sigma_h$ adopts a high value for consistent feedback (see Fig. 2). As soon as alternating feedback is given, the uncertainty, and thus the learning rate decreases to allow for refinements. The parameter $c_r$ denotes the constant rate and prevents stagnation in the event that $\sigma_{p,h} \approx 0$. GPC differs from COACH for updating the policy, since the error magnitude $e$ is now implicitly included in the computation of $r_k$ in (8).

An example of the policy and the learning rate during a learning process is depicted in Fig. 3. It shows an environment with a two-dimensional continuous state-space and an unstable equilibrium as reference at $(x_1, x_2) = (0, 0)$. The policy $(a)$ is trained by a teacher employing the ALR. The corresponding learning rate is displayed in $(b)$. Note that for critical states (area around $(x_1, x_2) = (0, 0)$) alternating feedback has caused the ALR to decrease, such that the policy can be refined.

3.3.2 Active Learning (AL)

The available uncertainty of the GPs can be used in an AL framework (Chernova and Thomaz 2014), where high-informative feedback can be queried for uncertain actions. Recent studies have shown great performance improvements with agent-induced feedback, mostly in the LfD domain (Gräve and Behnke 2013; Losey and O’Malley 2018). This study is, to the authors’ knowledge, the first to assess the potential with a directional feedback framework. Other than Chernova and Veloso (2009), who employed AL with the uncertainty of the visited state, we believe that especially the uncertainty in the action can advance the convergence of directive feedback methods. The motivation for this reasoning is that, in contrast to the LfD paradigm, inquiring human assistance in terms of feedback does not yield the optimal action instantaneously. In GPC - and feedback implementations in general - multiple feedback instances are needed to approach the optimal solution. Hence, rather than employing state uncertainty, we apply uncertainty per action, which is obtained by the human model

$$\Delta_k = c_o \sigma_h(z_k), \quad (9)$$

with $z_k$ the same as in (7) and gain $c_o$ to decouple AL from the ALR (see (8)). By inquiring feedback for high values of $\Delta_k$ we prioritize consistent feedback, since inconsistent feedback would reduce $\Delta_k$. AL will therefore
3.3.3 Sparsification for Corrective Learning

For every feedback instance provided by the trainer, the dictionary of the policy \( P \) is appended with the new tuple:

\[
N_p = \{ \ldots, (s_{m+1}, a_{m+1}) \} \tag{10}
\]

where \((s_{m+1}, a_{m+1})\) is calculated based on the executed action \( a_k \), learning rate \( r_k \) and feedback \( h_k \), i.e.

\[
a_{m+1} = a_k + r_k h_k.
\]

This approach renders the previous action \( a_k \) obsolete. In this application, a deficient property of GPs that hinders convergence is that by appending the dictionary following (10), the updated action on \( s_{m+1} \) is an average of \( a_k \) and \( a_{m+1} \) (assuming a coinciding or adjoining data instance). We therefore propose a sparsification method in which the tuple most relevant to the obsolete action \( a_k \) is omitted, rendering \( a_{m+1} \) the new action. Taking relevance into account while preserving the uncertainty estimation was not found in conventional online sparsification methods (e.g. \cite{Nguyen-Tuong2010}).

We therefore introduce a new sparsification technique that specifically applies to applications with iterative updates on the GP policy model.

The main outline of this sparsification is as follows: for every new feedback instance \((s_{m+1}, a_{m+1})\), the uncertainty of the policy \( \sigma_p(s_k) \) is compared against a certain threshold \( \sigma_{\text{thres}} \). We set this threshold to

\[
\sigma_{\text{thres}} = \frac{1}{2} \sqrt{\sigma_{s,m}^2},
\]

with \( \sigma_{s,m}^2 \) either from (4) or (5). In the event that this threshold is exceeded, the dictionary sample with the biggest covariance (i.e. smallest \( \text{Mahalanobis distance} \) \cite{Mahalanobis}) is omitted. We thereby prevent the policy from being negatively influenced by obsolete (old) training data. The sparsification method is presented in Algorithm 3.

Algorithm 3 Sparsification of policy \( P \) training data

```plaintext
1: function SPARS(s_k, a_m, \sigma_p, \sigma_{\text{thres}}, N_p)
2: if \sigma_p < \sigma_{\text{thres}} then
3:     index ← arg\max_i \ k_p(s_i, s_k)
4:     N_p(index) ← (s_k, a_n)
5: else
6:     Append dictionary \( N_p = \{ \ldots, (s_k, a_n) \} \)
7: end if
8: return \( N_p \)
9: end function
```

Further aid in establishing an inaccurate but rather complete policy as early as possible, before proceeding to the refinement stage.

4 EXPERIMENTAL SETUP

In this section we detail the experiments in which the performance of GPC is evaluated. The tests are carried out in three standardized benchmark problems from the OpenAI Gym \cite{Brockman2016}, namely the Inverted Pendulum(-v0), the Cart-Pole(-v0) and the Lunar Lander(-v2). The experiments with oracles (synthesized feedback source) are introduced to test the performance with consistency for all algorithms. The oracle tests also comprise the AL and ALR assessment. The applicability to the interactive domain is tested in separate experiments with actual human feedback. The performance of GPC\(^2\) will be tested against baseline COACH throughout \cite{Celemin2015}.

4.1 ORACLE TESTS

An oracle simulates human feedback based on a comparison of the executed action with a reference policy. The oracle experiments are carried out to exclude human factors such as inconsistency and limited attention that hinder a fair comparison between methods. Furthermore, it allows to accurately study the robustness of the algorithms to erroneous feedback.

4.1.1 Performance and Robustness tests

First, we will assess the performance of GPC for perfect and erroneous feedback. For the experiments we set a static feedback rate \( \gamma = 5\% \). When the action is close to the target action within the range \( \delta \), i.e. \( |a_k - a^*| = \delta \), the policy is considered converged and receives no more feedback. The robustness of the algorithms is tested by erroneous feedback. In this study, we will adopt error

\[1\]github.com/DWout/GPC
\[2\]github.com/rperezdattari/COACH-gym
rates of 10% and 20%, which will be administered following the protocol of Celemin et al. (2018).

4.1.2 Active Learning (AL)

The potential regarding AL is assessed by encouraging feedback for uncertainty policy actions. In order to exclude any random human factors, the performance is measured using an oracle. As such, we will adapt the feedback rate by incorporating the uncertainty of the human model $H$, i.e.: $\gamma = \Delta_k + \gamma_c$, with $\Delta_k$ as in $\Delta$ and $\gamma_c$ denoting the minimum feedback rate. The application of AL is measured against a baseline with the same episodic feedback rate, but not uncertainty triggered (see Table [1] $ii$ and $iv$). The ALR is excluded from the test to circumvent any influences. To account for feedback inconsistencies the erroneous feedback likelihood is set to 10%.

4.1.3 Ablation Study

The ablation assessment will analyze the contribution of the ALR (Section 4.1.2). We will run oracle tests employing the learning rate in $\Delta$ and compare this to the baseline test from Section 4.1.2 with the same episodic feedback rate and erroneous feedback likelihood. In addition, we will test a combination of AL and ALR. A summary of the experimental setup is presented in Table [1].

Table 1: Overview of the experiments regarding Active Learning (AL) and the ablation study for the Adaptive Learning Rate (ALR). For fair comparison the experiments are conducted with the same feedback (Fb) rate $\gamma$.

| AL: | ALR: | Fb rate $\gamma$: | Learning rate $r_k$: |
|-----|------|-----------------|-----------------|
| $i$ | ✓    | $\Delta_k + \gamma_c$ | $\sigma_p(z_k) + \sigma_h(z_k) + c_r$ |
| $ii$ | ✓    | $\Delta_k + \gamma_c$ | $r_c$ |
| $iii$ | ✗    | $\gamma_{exp}(t)$ | $\sigma_p(z_k) + \sigma_h(z_k) + c_r$ |
| $iv$ | ✗    | $\gamma_{exp}(t)$ | $r_c$ |

4.2 HUMAN TEACHERS

This section will elaborate the experiments for validating the application of GPC to interactive settings. The experiments are conducted by employing four human teachers (in the age of 20 to 30, of different background) to the three proposed benchmarks with the objective to achieve convergence as fast as possible. The participants perform the training with every algorithm for every environment four times: two dummy runs to get acquainted with the environment, and two real runs that are recorded. The tests runs are performed single blind: the participants are not informed about which algorithm they controlled.

5 EXPERIMENTAL RESULTS

In this section, we report GPC’s performance on the three domains (see Section 4.2). The kernels and hyperparameters for the human model and the policy are depicted in Table 2. For readability purposes the presented results are a walking mean of 3 samples, unless otherwise specified.

Table 2: Hyperparameters of the GPs in the benchmarks. The policy and human model are modelled by Squared Exponential (SE) and Matérn (Mat.) kernel. The constant learning rate in $\Delta$ is denoted as $c_r$.

| Pendulum Cart-Pole Lunar Lander | CS | NS | CS | NS | CS | NS | CS | NS |
|-------------------------------|----|----|----|----|----|----|----|----|
| $H$: SE                       | 0.7 | 0.5 | 0.1 | 0.6 | 0.01 | 0.01 | 0.01 | 0.01 |
| $I_p$: Mat. Mat. Mat. Mat. Mat. | 0.03 | 0.07 | 0.5 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| $\nu_p$: Mat. Mat. Mat. Mat. Mat. | 0.05 | 0.5 | 0.0 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| $c_r$: Mat. Mat. Mat. Mat. Mat. | 0.5 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |

5.1 PERFORMANCE AND ROBUSTNESS

The return for the Pendulum domain is depicted in Fig. 5. The GPC variants show similar convergence and robustness properties. Due to the coarser exploration in the initial learning phase the learning curve is steeper in comparison to COACH. The final performance is similar. The average learning rate for all consecutive feedback instances is depicted in Fig. 6 for an error rate of 0% and 20%. For GPC we see a more aggressive learning rate for the initial learning phase, which diminishes upon...
The steeper initial learning curve, which was observed in the Pendulum domain in Fig. 5 also distinguishes GPC from COACH in the Cart-Pole environment (see Fig. 7). The protracted take-off time for COACH is presumably a result of the human feedback supervised learner module (see Celemin and Ruiz-del Solar, 2015) that adopts a reduced learning rate for the initial learning phase. For every error rate GPC outperforms COACH. The tuning convenience for GPC-NS shows to trade with some suboptimal performance for erroneous feedback.

The performance of GPC and COACH in the Lunar Lander domain is depicted in Fig. 8. Both GPC implementations achieve good performance either for ideal or erroneous feedback. COACH yields poor performance due to intractability of the feature space, which is custom parameterized in $\mathbb{R}^{24}$.

Figure 6: Normalized average learning rate in the Pendulum-v0 domain for both GPC and COACH. In contrast to existing methods, the learning rates of the GPC implementations diminish over time, such that the corrections become more subtle upon convergence. As desired, this reduction develops more gradually in case of erroneous feedback.

Figure 7: Average return per episode for the CartPole-v0 domain. Both GPC implementations outperform COACH for ideal feedback. GPC shows good robustness to erroneous feedback, whereas GPC-NS is more brittle.

The average learning rate for the ALR tests measures 0.0386 for i and 0.0374 for iii, which is lower on average but better balanced to the static rate of $r_c = 0.4$ in ii and iv.

Figure 8: Average return for the LunarLander-v2 environment. Both GPC implementations achieve good performance either for ideal or erroneous feedback. COACH yields poor performance due to intractability of the feature space, which is custom parameterized in $\mathbb{R}^{24}$.

5.2 ACTIVE LEARNING AND ABLATION

The result of the four test cases from Table 1 are displayed in Fig. 9 for the Cart-Pole environment with constant feedback likelihood of $\gamma_c = 0.01$, constant additive learning rate of $c_r = 0.01$ and a static learning rate of $r_c = 0.4$. AL combined with ALR has superior performance. The individual components (ALR and AL resp.) both prove their significance compared to the baseline. The average learning rate for the ALR tests measures 0.0386 for i and 0.0374 for iii, which is lower on average but better balanced to the static rate of $r_c = 0.4$ in ii and iv.

5.3 HUMAN VALIDATION

The performance for all environments is depicted in Fig. 10. For the Inverted Pendulum and the Cart-Pole environment both COACH and GPC converge to maximal performance. Although some relative differences are noticeable in the learning curve, the variations are not deemed statistically significant considering the number of tests. The fact that the lacking robustness of the COACH implementation (Fig. 7) does not emerge in this result is notable. In contrast to the static behavior of oracles, humans anticipate to the consequences of the provided feedback and adapt their feedback strategy accordingly. When the corrections at a particular state are deemed insufficient, teachers may choose to provide multiple feedback samples subsequently in order to realize the intended effect. An interesting observation is
Figure 9: Average return (a) and feedback rate (b) for the Active Learning (AL) and the ablation of Adaptive Learning Rate (ALR) in the Cart-Pole domain. AL and ALR combined achieve superior performance. It shows that the Adaptive Learning Rate accelerates the convergence with less feedback.

the difference in return for the Lunar Lander environment in (c), which validate the findings from the oracle benchmarks in Fig. 8. The unfeasible parameterization of the feature space severely deteriorate the performance in higher dimensional problems.

6 CONCLUSION

Humans are very efficient in understanding control strategies using intuition and common sense. Corrective feedback is an especially effective means of communication and the current state-of-the-art, COrrective Advice Communicated by Humans (COACH), enables one to establish a control law without requiring control or engineering expertise. Moreover, performance is superior over methods that learn autonomously or from evaluative feedback. However, COACH employs Radial Basis Function (RBF) networks for modelling which requires meticulous feature space engineering before these advantages enter into force.

In this work, we have presented GPC. It has an architecture similar to COACH, but it engages Gaussian Processes (GPs) such that modelling expertise is no longer required and the limitation to confined problems is hereby overcome. Moreover, we leverage the available uncertainty with an Adaptive Learning Rate (ALR) that adapts to the trainer’s learning phase. In addition, we introduced a new sparsification technique, specifically designed for efficient and accelerated GP policy updates. GPC was applied to three continuous benchmarks from the OpenAI Gym: the Inverted Pendulum, Cart Pole, and Lunar Lander. Our novel framework outperforms COACH on every domain tested by means of faster learning and better robustness to erroneous feedback. The greatest improvement was for the Lunar Lander problem, where RBF parameterization fails but GPC is flawless.

In addition to the performance and robustness assessment, we performed two additional studies: 1) We have addressed the potential of Active Learning (AL) and demonstrated how eliciting feedback for actions with greatest uncertainty yields drastic improvements on convergence. 2) We have furthermore presented an alternative implementation GPC-NS where length-scale tuning is circumvented by online normalization of the input space. It is slightly suboptimal and trades some robustness in comparison to GPC, but a great advantage is that it does not require any input parameterization in new domains. This could especially be decisive in higher-dimensional problems, and furthermore renders our work feasible also for non-experts.

In future work, it might be possible to further innovate the dynamical scaling, such that the applicability and generality of GPC-NS is again extended. In addition, the AL opportunities assessed here deserve further research and should be validated also with human participants.

References

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In International Conference on Machine Learning (ICML).

Abramowitz, M. and Stegun, I. A. (1965). Handbook of Mathematical Functions: With Formulas, Graphs,
Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). OpenAI gym. arXiv:1606.01540 [cs.LG].

Busoniu, L., Babuska, R., De Schutter, B., and Ernst, D. (2010). *Reinforcement Learning and Dynamic Programming using Function Approximators*. CRC press.

Busoniu, L., Babuska, R., De Schutter, B., and Ernst, D. (2010). COACH: Learning continuous actions from corrective advice communicated by humans. In *IEEE International Conference on Advanced Robotics (ICAR)*, pages 581–586.

Celemin, C., and Ruiz-del Solar, J. (2015). COACH: Learning continuous actions from corrective advice communicated by humans. In *IEEE International Conference on Advanced Robotics (ICAR)*, pages 581–586.

Celemin, C., Ruiz-del Solar, J., and Kober, J. (2018). A fast hybrid reinforcement learning framework with human corrective feedback. *Autonomous Robots*, First Online.

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

Chernova, S. and Veloso, M. (2009). Interactive policy learning through confidence-based autonomy. *Journal of Artificial Intelligence Research*, 34:1–25.

Duvanau, D. (2014). *Automatic Model Construction with Gaussian Processes*. PhD thesis, University of Cambridge.

Gräve, K. and Behnke, S. (2013). Learning sequential tasks interactively from demonstrations and own experience. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3237–3243.

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

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. (2018). Rainbow: Combining improvements in deep reinforcement learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 6, pages 1000–1005.

Knox, W. B. and Stone, P. (2009). Interactively shaping agents via human reinforcement: The TAMER framework. In *International Conference on Knowledge Capture*, pages 9–16.

Losey, D. P. and O’Malley, M. K. (2018). Including uncertainty when learning from human corrections. arXiv:1806.02454 [cs.RO].

Maeda, G., Ewerton, M., Osa, T., Busch, B., and Peters, J. (2017). Active incremental learning of robot movement primitives. In *Annual Conference on Robot Learning (CoRL)*, pages 37–46.

Mahalanobis, P. C. (1936). On the generalized distance in statistics. volume 2, pages 49–55. National Institute of Science of India.

Neal, R. M. (1995). *Bayesian learning for neural networks*. PhD thesis, University of Toronto.

Nguyen-Tuong, D. and Peters, J. (2010). Incremental sparsification for real-time online model learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 557–564.

Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian Processes for Machine Learning*, volume 2. MIT Press Cambridge, MA.

Ross, S., Gordon, G., and Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 627–635.

Suay, H. B. and Chernova, S. (2011). Effect of human guidance and state space size on interactive reinforcement learning. In *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*.

Thomaz, A. L. and Breazeal, C. (2006). Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 6, pages 1000–1005.