Interactive Robot Education

Riad Akrour, Marc Schoenauer, Michèle Sebag

Interactive Robot Education

Riad Akrour, Marc Schoenauer, and Michèle Sebag

TAO, CNRS − INRIA − LRI
Université Paris-Sud, F-91405 Orsay Cedex
FirstName.Name@lri.fr

Abstract. Aimed at on-board robot training, an approach hybridizing active preference learning and reinforcement learning is presented: *Interactive Bayesian Policy Search* (IBPS) builds a robotic controller through direct and frugal interaction with the human expert, iteratively emitting preferences among a few behaviors demonstrated by the robot. These preferences allow the robot to gradually refine its policy utility estimate, and select a new policy to be demonstrated, after an Expected Utility of Selection criterion.
The paper contribution is on handling the preference noise, due to expert's mistakes or disinterest when demonstrated behaviors are equally unsatisfactory. A noise model is proposed, enabling a resource-limited robot to soundly estimate the preference noise and maintain a robust interaction with the expert, thus enforcing a low sample complexity. A proof of principle of the IBPS approach, in simulation and on-board, is presented.

1 Position of the problem

Reinforcement learning (RL) [5, 19, 22], aimed at optimal sequential decision making, has been particularly investigated in robotics (see [12] for a survey). In practice, its success critically depends on the smart design of i) the state and action spaces; ii) the reward function. This paper focuses on the reward shaping issue: Not only should the reward function reflect the target robot behavior; it should also induce a robust and tractable optimization problem. Some approaches, based on the demonstration of the target behavior by the expert and ranging from inverse reinforcement learning [17], to learning by imitation [6] or learning by demonstration [14], have been proposed to learn an appropriate reward function. Such approaches however require a strong expertise in the solution of the task at hand. Alleviating this requirement, alternative approaches have been proposed since the late 2000s, based on the expert's preferences about the robot demonstrations, and automatically deriving a reward function [7], a posterior over the parametric policy space [24] or a policy utility estimate [2, 3].

The general limitations of preference-based RL are twofold. On the one hand, it should require *little* feedback from the expert (of the order of a few dozen preference judgments) to be effectively usable. On the other hand, it must resist the preference noise due to expert's actual mistakes or disinterest, e.g. when demonstrated robot behaviors are equally unsatisfactory.

The contribution of the paper, the *Interactive Bayesian Policy Search* (IBPS) framework, simultaneously estimates the policy utility and the confidence thereof, through modelling the expert preference noise. The goal is to decrease the sample complexity of the interactive robot education process, that is the number of preference judgments required to reach a satisfactory behavior. This goal is achieved *on-board* through a more robust selection of the policies to be demonstrated.

Notations. The standard notations of Markov Decision Processes are used in the rest of the paper [19, 22]. Inverse reinforcement learning (IRL), a.k.a. learning by imitation or apprenticeship learning, considers an MDP$\backslash r$ (reward is unknown), a set of state sequences $c_k = (s_0^{(k)}\ s_1^{(k)}\ s_2^{(k)} \dots s_{T_k}^{(k)}))$ describing an expert trajectory, and a feature function $\phi(.)$ mapping the states (and hence the trajectories) onto $\mathbb{R}^d$, with $\mathbf{u}(c_k) = \sum_{h=0}^{\infty} \gamma^h \phi(s_h^{(k)})$, where $0 < \gamma < 1$ is the discount factor.

Assuming that the expert trajectories maximize some target reward r^*, the IRL goal is to find a policy with quasi-optimal performance under r^* (rather than finding r^* itself) [1]. Specific exploration mechanisms, e.g. based on Gibbs sampling [13], are used to overcome under-optimal expert trajectories. [11] combines classification-based RL [15, 14] and IRL, formalizing IRL as a structured classification problem.

Preference learning-based RL was designed to alleviate the strong expertise assumption underlying the exploitation of human demonstrated trajectories. In [7] the authors use preference learning in replacement of classification algorithms in rollout classification-based policy iteration (RCPI) [9]. The authors advocate that action ranking is more flexible and robust than a supervised learning based approach. In [2], the authors use preference learning to define a policy utility estimate; the main difference with [7] is that the former defines an order relation on the action space depending on the current state, whereas the later defines an order relation on the trajectories. Active preference-based policy learning [24, 3] exploits pairwise preferences among trajectories demonstrated by the agent, whereas IRL exploits trajectories demonstrated by the human expert. The main difference between [24] and [3] concerns the demonstrated trajectories: short trajectories are selected in [24], assuming that the initial states can be sampled after some prior distribution $P(s_0)$ in order to enforce visiting interesting regions of the state space. No such selection of interesting excerpts is required in [3]; in counterpart, the expert judgment errors might become more likely as the expert is required to see the whole trajectories.

Formally, [24] infers a distribution over the parametric policy space. On the one hand, this setting enables the direct sampling of policies from the posterior. On the other hand, it makes it more expensive to update the posterior distribution after a preference constraint is added: the similarity between a parametric policy and the trajectories involved in a preference constraint requires a good amount of rollouts of the parametric policy to be computed. The policy selection in [3] is inspired from the expected utility of selection principle [23]. For tractability, a criterion based on the ranking-SVM objective value is used to estimate the utility of selecting a policy, conditionally to it being better/worse

than the current best policy. However, the preference judgment noise is not taken into account.

The IBPS approach, building upon the APRIL framework [3], focusses on the policy robustness w.r.t. noisy expert preferences (section 2). A noise model is estimated on-board, enabling a limited-resource robot to cope with less-than-ideal experts. Under this model, the expected utility of selection is proved to be a good approximation of the expected posterior utility, the ideal but hardly tractable active selection criterion. The in-situ evaluation of IBPS establishes a proof of principle of the approach and sheds some light on the expert judgment errors (section 3). Their impact on the robot training (with and without noise modelling) is discussed in section 4.

2 Interactive Bayesian Policy Search

The IBPS algorithm elaborates on an active preference policy learning setting, inspired from [3] and distinguishing two search spaces. The first space $\mathcal{X}$, referred to as *input or parametric space*, is used to describe and sample policies (in the following $\mathcal{X} = \mathbb{R}^d$). Policy $\pi_{\mathbf{x}}$ is represented as a vector $\mathbf{x}$, describing the state-action mapping (subscript $\mathbf{x}$ will be omitted for readability when clear from the context). The second space $\Phi(\mathcal{X})$ and referred to as *feature or behavioral space*, is used to describe a policy behavior or trajectory, and learn the policy utility function. Formally, a robot trajectory generated from policy $\pi_{\mathbf{x}}$, expressed as a state-action sequence $(s_0, \pi_{\mathbf{x}}(s_0), s_1 \ldots \pi_{\mathbf{x}}(s_{H-1}), s_H)$ is represented as a vector $\mathbf{u} \in \mathbb{R}^D$, with $\mathbf{u} = \sum_{i=0}^{H} \gamma^i \phi(s_i)$, $0 < \gamma \leq 1$ and ϕ a feature function mapping the state space (or the state $\times$ action space) onto $\mathbb{R}^D$. In the following, we shall restrict ourselves to linear trajectory utilities, where the utility of trajectory $\mathbf{u}$ is set to $\langle \mathbf{w}, \mathbf{u} \rangle$ and $\mathbf{w}$ is a unit vector in $\mathbb{R}^D$. The $\pi_{\mathbf{x}}$ policy utility is the trajectory utility expectation, over the trajectories generated from $\pi_{\mathbf{x}}$.

Following the general framework described in [3, 24], active preference-based policy learning iterates a 3-step process: i) the learning agent demonstrates at least one new trajectory; ii) the expert expresses pairwise preferences between two newly demonstrated trajectories or a new trajectory and the former best trajectory (or his recollection thereof); iii) the policy utility estimate, that is the robot model of expert's preferences, is updated and a new policy is selected according to an active criterion.

After [3], the use of both the input/parametric and the feature/behavioral spaces is motivated by the expressiveness/tractability dilemma. On the one hand, a high dimensional continuous search space is required to express competent policies. Still, such high-dimensional search spaces makes it difficult to learn a preference-based policy return from a moderate number of expert preferences. On the other hand, the behavioral space does enable to learn a preference-based policy return from the little evidence provided by the expert, despite the fact the behavioral description is insufficient to describe a flexible policy.

2.1 Preference and noise model

Letting $\mathbf{w}^*$ denote the true (hidden) utility of the expert, his preference judgment on the pair of trajectories $\{\mathbf{u}, \mathbf{u}'\}$ is modeled as a noisy perturbation of his true preference $\langle \mathbf{w}^*, (\mathbf{u} - \mathbf{u}') \rangle$. The preference noise is usually modelled in the literature as a Gaussian perturbation ([8, 24]) or following the Luce-Shepard model [16, 18, 23]. In both cases, the noise model involves an extra-parameter (respectively the standard deviation of the Gaussian perturbation or the temperature of the Luce-Shepard rule) controlling the magnitude of the noise.

The noise model considered in IBPS involves a single scalar parameter $\delta \in \mathbb{R}, \delta > 0$, where the probability of the expert preferring $\mathbf{u}$ over $\mathbf{u}'$ given the true preference $\mathbf{z} = \langle \mathbf{w}^*, (\mathbf{u} - \mathbf{u}') \rangle$ is defined as: $P(\mathbf{u} \succ \mathbf{u}' \mid \mathbf{w}^*, \delta) = \frac{1}{2\delta}\mathbf{z} + \frac{1}{2}$, if $|\mathbf{z}| < \delta$ and $P(\mathbf{u} \succ \mathbf{u}' \mid \mathbf{w}^*, \delta) = 1$ (resp. 0) if $\mathbf{z} \geq \delta$ (resp. $\mathbf{z} \leq -\delta$).

This simple ridged model allows IBPS to handle the uncertainty on the noise threshold δ through analytical integration over the δ distribution. A first option is to consider that the expert consistently answers according to a hidden but fixed δ^*. The second option assumes that δ can vary over time. Arguably, the former option is less flexible and less robust, as one abnormally large mistake can prevent the robot from identifying an otherwise consistent ranking model. Inversely, the latter option while being more robust could slow down the identification of the expert's utility function. This latter option is however clearly more appropriate to accommodate the cases where the task (or the expert's understanding thereof, or his preferences) might change over time. In the remainder of the paper, a distinct noise scale parameter δ_t is considered to model the expert preference noise after the t-th preference judgment has been emitted.

Let $\mathcal{U}_t = \{\mathbf{u}_0, \mathbf{u}_1, \ldots; (\mathbf{u}_{i_1} \succ \mathbf{u}_{i_2}), i = 1 \ldots t\}$ denote the archive of all trajectories demonstrated to the expert and the expert's preference judgments up to the t-th iteration. Let us set a uniform prior $p(w)$ over the unit sphere $\mathbf{W}$ of $\mathbb{R}^D$, and let us likewise assume that the prior over the noise scale parameter δ_i is uniform on interval $[0, M]$. Given $\mathcal{U}_t$, the posterior distribution of the utility function reads:

$$p(\mathbf{w}; \mathcal{U}_t) \propto \prod_{i=1,t} P_N(\mathbf{u}_{i_1} \succ \mathbf{u}_{i_2} \mid \mathbf{w}) = \prod_{i=1,t} \left(\frac{1}{2} + \frac{\mathbf{z}_i(\mathbf{w})}{2M} \left(1 + \log \frac{M}{\mid \mathbf{z}_i(\mathbf{w}) \mid} \right) \right) \tag{1}$$

where $\mathbf{z}_i$ is set to $\langle \mathbf{w}, (\mathbf{u}_{i_1} - \mathbf{u}_{i_2}) \rangle$ if it is not greater (resp. lower) than M (resp. $-M$) in which case it takes the value M (resp. $-M$); $P_N(\mathbf{u}_{i_1} \succ \mathbf{u}_{i_2} \mid \mathbf{w})$ is set to $\int_0^{+\infty} P(\mathbf{u}_{i_1} \succ \mathbf{u}_{i_2} \mid \mathbf{w}, \delta)\, \mathrm{d}\delta$ (where P_N stands for *noisy* ranking probability). When $\mathbf{z}_i(\mathbf{w}) = 0, P_N(\mathbf{u}_{i_1} \succ \mathbf{u}_{i_2} \mid \mathbf{w}) = 1/2$. Note that P_N is a sigmoid only depending on the utility difference $\langle \mathbf{w}, (\mathbf{u} - \mathbf{u}') \rangle$, thus very similar to the Gaussian noise model[1]. However it is amenable to formal analysis (section 2.2) and its parameterization is more intuitive, as the upper bound M on the utility margin subject to preference noise is expressed using the same unit as the utility itself. For instance in the case where $\mathbf{u}$ and $\mathbf{u}'$ are state histograms with

[1] Reading $P_{Gauss}(\mathbf{u} \succ \mathbf{u}' \mid \mathbf{w}^*, \delta) = \frac{1}{2} + \frac{1}{2}\mathrm{erf}(\frac{\langle \mathbf{w}^*, \mathbf{u} - \mathbf{u}' \rangle}{\sqrt{2}\delta})$

L_1 norm 1, setting $M = \sqrt{2}/2$ guarantees that if $|\langle \mathbf{w}, \mathbf{u} - \mathbf{u}' \rangle|$ is more than half the maximum utility difference, these cannot be mistakingly ordered (since $\max_{\mathbf{w},\mathbf{u},\mathbf{u}'} |\langle \mathbf{w}, (\mathbf{u} - \mathbf{u}') \rangle| \leq \max_{\mathbf{w},\mathbf{u},\mathbf{u}'} ||w||_2 \, ||\mathbf{u} - \mathbf{u}'||_2 \leq \sqrt{2} \, ||\mathbf{u} - \mathbf{u}'||_\infty \leq \sqrt{2}$).

2.2 Active policy selection

Given a trajectory $\mathbf{u}$, its expected utility $\mathrm{EU}_N(\mathbf{u}; \mathcal{U}_t)$ reads $\int_W \, p(\mathbf{w}; \mathcal{U}_t) \, \langle \mathbf{w}, \mathbf{u} \rangle \mathrm{d}\mathbf{w}$ where p stands for the posterior $\mathbf{w}$ distribution on the unit sphere $\mathbf{W}$. By construction, the most informative pair of trajectories to submit to the expert's judgment is $\{\mathbf{u}_1, \mathbf{u}_2\}$ with maximum expected *posterior* utility:

$$\mathrm{EPU}_N(\{\mathbf{u}_1, \mathbf{u}_2\}; \mathcal{U}_t) = \sum_{i=1,2} P_N(\mathbf{u}_i \succ \mathbf{u}_{3-i} \mid \mathcal{U}_t) \; \mathrm{EU}^*(\mathcal{U}_t \cup \{\mathbf{u}_i \succ \mathbf{u}_{3-i}\}) \quad (2)$$

where $P_N(\mathbf{u} \succ \mathbf{u}' \mid \mathcal{U}_t) = \int_W P_N(\mathbf{u} \succ \mathbf{u}' \mid \mathbf{w}) p(w; \mathcal{U}_t) \mathrm{d}\mathbf{w}$ and $\mathrm{EU}^*(\mathcal{U}_t) = \max_u \mathrm{EU}(\mathbf{u}; \mathcal{U}_t)$.
This selection criterion ensures that the selected pair alters the posterior distribution in such a way that it maximizes the EU one step ahead. After [23], in the noiseless case (subscript NL) it is equivalent to maximize the expected utility of selection and the expected posterior utility ($\mathrm{EUS}^*_{NL} = \sum_{i=1,2} P_{NL}(\mathbf{u}_i \succ \mathbf{u}_{3-i} \mid \mathcal{U}_t) \; \mathrm{EU}(\mathbf{u}_i; \mathcal{U}_t \cup \{\mathbf{u}_i \succ \mathbf{u}_{3-i}\}) = \mathrm{EPU}^*_{NL}$, where EPU_{NL} differs from EPU_N as P_N is replaced with $P_{NL}(\mathbf{u} \succ \mathbf{u}' \mid \mathbf{w})$, set to 1 if $\mathbf{z} = \langle \mathbf{w}, (\mathbf{u} - \mathbf{u}') \rangle > 0$, 1/2 if $\mathbf{z} = 0$ and 0 otherwise.

The above result establishes the computational tractability of the approach in the noiseless case, as no optimization resolution is required to compute EUS as opposed to EPU. Specifically in the considered context, using EUS instead of EPU saves the resolution of two control problems for every evaluation of a pair of trajectories.

A first contribution of the present paper is to show that in the noisy case, and when considering the above noise model, the optimal EUS is upper-bounded by the optimal EPU less a constant term: $\mathrm{EUS}^*_N \leq \mathrm{EPU}^*_N - L_N$, where L_N is a positive constant only depending on the noise hyper-parameter M.

Lemma 1 For any set $q = \{\mathbf{u}, \mathbf{u}'\}$ of two trajectories $\mathbf{u}$ and $\mathbf{u}'$, and for any archive $\mathcal{U}_t$, we have:

$$\mathrm{EUS}_{NL}(q, \mathcal{U}_t) - \frac{M}{2\lambda}(1 - \frac{1 + ln\lambda}{\lambda}) \leq \mathrm{EUS}_N(q; \mathcal{U}_t) \leq \mathrm{EUS}_{NL}(q, \mathcal{U}_t)$$

With $\lambda = e^{-\frac{1}{2} - W_{-1}(-\frac{1}{2} e^{-\frac{1}{2}})}$ and W_{-1} is the lower branch of the Lambert W function. Let $L_N = \frac{M}{2\lambda}(1 - \frac{1+ln\lambda}{\lambda}) \approx \frac{M}{19.6433}$

The lower bound is tight but corresponds to a degenerate case where $p(\mathbf{w}; \mathcal{U}_t)$ is concentrated on a set of $\mathbf{w}$ such that $|\mathbf{w} \cdot (\mathbf{u} - \mathbf{u}')|$ is neither too small in which case the probability of preferring the wrong object is high but the loss in utility is small, nor too large translating in a high but unlikely loss in utility. The upper

bound holds for any noise model. There is equality between EUS_N and EUS_{NL} when $|\mathbf{w} \cdot (\mathbf{u} - \mathbf{u}')| \geq M$ for all $\mathbf{w}$ s.t. $p(\mathbf{w}; \mathcal{U}_t) > 0$. Due to space constraints, proofs are provided in the supplementary material.

Let $\text{EUS}^*_N(\mathcal{U}_t) = \max_{u,u'} \text{EUS}_N(\{\mathbf{u}, \mathbf{u}'\}; \mathcal{U}_t)$ and $\text{EPU}^*_N(\mathcal{U}_t) = \max_{u,u'} \text{EPU}_N(\{\mathbf{u}, \mathbf{u}'\}; \mathcal{U}_t)$. Using Lemma 1 and results from [23] it immediately follows that:

Theorem 1 $\text{EUS}^*_N(\mathcal{U}_t) \geq \text{EPU}^*_N(\mathcal{U}_t) - L_N$

Since EPU is always greater than EUS and in particular for the set $\{\mathbf{u}^*, \mathbf{u}'^*\}$ maximizing the EUS, this theorem unsure that the set $\text{EPU}_N(\{\mathbf{u}^*, \mathbf{u}'^*\}; \mathcal{U}_t)$ is no lower than L_N from $\text{EPU}^*_N(\mathcal{U}_t)$.

2.3 Optimization of the active selection criteria

In the direct policy search context, the optimization of the EUS criterion raises two main issues as the policy search operates in the input space $\mathcal{X}$, while the utility criterion is defined on the feature space $\Phi(\mathcal{X})$. The first issue results from the stochastic input-to-feature space mapping due to e.g. environment noise: while selecting a pair of trajectories with high EUS, these trajectories cannot always be demonstrated verbatim to the expert, possibly resulting in poorly informative trajectory demonstrations[2]. The latter issue regards the optimization of the EUS criterion on the feature space, exploring a large search space $\mathcal{X}$, thus entailing potentially prohibitive computational costs.

In [24], both issues are dealt with by assuming a pool of trajectories, sampled from the current posterior over the parametric space, to be recorded beforehand in each iteration. The selected trajectories can thus be demonstrated *verbatim* to the expert. In [3], the pair of trajectories to be compared always includes the best trajectory so far noted $\mathbf{u}_t$. The policy π to be demonstrated is selected by maximizing the approximate EUS, averaging the $\text{EUS}(\mathbf{u}, \mathbf{u}_t)$ over a few trajectories $\mathbf{u}$ generated from π.

Forcing $\mathbf{u}_t$, the best so far trajectory in the archive $\mathcal{U}_t$, as one of the two trajectories of the next preference query has several benefits. Firstly, it cuts the burden on the human expert by half: a single new trajectory needs be demonstrated in each iteration as $\mathbf{u}_t$ is known to the expert. Secondly, the optimization problem on $\mathcal{X} \times \mathcal{X}$ reduces to a more tractable optimization problem on $\mathcal{X}$. In counterpart, it entails the loss of optimality of the selection process (since the optimal query pair does not necessarily contains $\mathbf{u}_t$). A further justification why the selection of $\mathbf{u}_t$ as reference term of the preference query might work well in practice, comes from the fact that EUS_N approximates EUS_{NL} up to L_N and EUS_{NL} is a sub-modular function: after [23], selecting a query pair x_1, x_2 by greedily selecting x_1 with highest expected utility (EU) and x_2 the one maximizing EUS_{NL} w.r.t. x_1, results in $\text{EPU}_{NL}(x_1, x_2) \geq .75\text{EPU}^*_{NL}$. Indeed, $\mathbf{u}_t$ has a high expected utility in the noiseless case; the probability distribution on

[2] This limitation can be alleviated through recording the trajectories considered by the robot on its own, and showing the most informative one to the expert, as in [24].

the utility functions concentrates on $\mathbf{w}$ such that it ranks $\mathbf{u}_t$ above all other trajectories in $\mathcal{U}_t$.

The former issue, related to the stochastic input-to-feature space mapping, is circumvented in the case of an ergodic MDP when considering long trajectories relatively to the policy mixing time. These conditions are however hard to meet in practice. A fall back procedure consists of selecting policy π which maximizes an empirical estimate of $\mathbb{E}_{u\sim\pi}\mathrm{EUS}_N(\mathbf{u};\mathbf{u}_t,\mathcal{U}_t)$, where trajectories $\mathbf{u}$ are generated after π.In the noiseless case, a lower bound on the above criterion is given by $\mathrm{EUS}_{NL}(\mathbb{E}_{u\sim\pi}(\mathbf{u});\mathbf{u}_t,\mathcal{U}_t)$ (due to the convexity of the max operator on $\mathbf{W}$ and the Jensen inequality). The optimization of this lower bound only requires to integrate over $\mathbf{W}$ once, thus with tractable cost.

In the noisy case, the optimization of the $\mathrm{EUS}_N(\mathbf{u};\mathbf{u}_t,\mathcal{U}_t)$ criterion can hardly be handled through RL methods as the criterion is not linear in $\mathbf{u}$. The approach used to tackle this optimization problem is based on the resolution of a sequence of linear optimization problems[3]. Formally, a sequence of linear problems aimed at optimizing EUS_{NL} is considered to find an approximate optimum of EUS_N. At each iteration, RL is used to find a policy that will increase EUS_{NL} until reaching a fixed point. The algorithm iteratively proceeds as follows: i) an initial utility $\mathbf{w}_0$ is sampled from the posterior $p(\mathbf{w};\mathcal{U}_t)$; ii) at the k-th iteration RL is used to find $\pi_k = \arg\max_\pi \mathbb{E}[\langle \mathbf{w}_k, \sum_{i=0}^{i=\infty}\gamma^i\phi(s_i) \mid s_{i+1}\sim p(s_i,\pi(s_i))\rangle]$; iii) the average trajectory $\bar{\mathbf{u}}_k$ associated to policy π_k is determined; iv) a new utility function $\mathbf{w}_{k+1} = \int_W p(\mathbf{w}\mid\mathcal{U}_t)1_{\{\langle w,\bar{u_k}-u_t\rangle>0\}}\mathrm{d}\mathbf{w}$ is generated; v) k is incremented until a stopping criterion is reached (see below).

The convergence of the above iterative process follows from its monotonicity: it is shown that (see supplementary material)

$$\mathrm{EUS}_{NL}(\bar{\mathbf{u}}_0;\mathbf{u}_t,\mathcal{U}_t) \leq \mathrm{EUS}_{NL}(\bar{\mathbf{u}}_1;\mathbf{u}_t,\mathcal{U}_t) \leq \cdots \leq \mathrm{EUS}_{NL}(\bar{\mathbf{u}}_{k+1};\mathbf{u}_t,\mathcal{U}_t)$$

The quality of the local optimum $(\mathbf{w}_k,\pi_k)$ thus discovered is improved through restarting the process from diverse initial $\mathbf{w}_0$ independently sampled from $p(\mathbf{w};\mathcal{U}_t)$. It is worth noting that albeit the optimization considers the noiseless EUS_{NL} (which is according to Lemma 1 at most L_N far from EUS_N), the noise in expert preferences is still taken into account when updating the posterior. This iterative algorithm is similar to the Query Iteration [23], augmented with an RL step.

The search for π_k given $\mathbf{w}_k$ proceeds as follows. When considering a discrete state space of restricted size, the canonical feature function measures the visiting frequency of each state[4] ($\phi_j(s^i)=\delta_{i,j}$ with $\delta_{i,j}=1$ iff $i=j$ and 0 otherwise). In

[3] Another approach, based on *Direct Policy Search* (DPS), would be to find $\arg\max_\pi \mathbb{E}_{u\sim\pi}\mathrm{EUS}_N(\mathbf{u};\mathbf{u}_t,\mathcal{U}_t)$, using e.g. [10]. Interestingly, such an approach could accommodate finite horizon and does not require the linearity w.r.t. $\mathbf{u}$; non-linear feature functions ψ could thus be considered, extending the set of policies learnable with IBPS. The investigation of the DPS approach, and its comparison with the current IBPS approach, is left for further study.

[4] In the general case of a feature function on a discrete state space, a change of representation on the feature space can be used to get back to the canonical feature function case; see supplementary material.

this case, the i-th coordinate w_i of the utility can be interpreted as the reward associated to s_i and standard RL algorithms such as Policy Iteration can be used to find π_k from $\mathbf{w}_k$. In the case of a large or continuous state space, state of the art approximate RL algorithms such as LSTD or GTD [15, 20] can be used[5].

2.4 Estimation of the active selection criteria

$\mathrm{EUS}(\mathbf{u}; \mathbf{u}_t, \mathcal{U}_t)$ involves computing an integral over the hypersphere $\mathbf{W}$ weighted by the posterior $p(w; \mathcal{U}_t)$. It is approximated using importance sampling, first generating n samples uniformly in $\mathbf{W}$, and updating their weight after each update of the archive. When, at some k-th iteration, the number of effective particles $1/\sum_{i=1}^{n} w_i^2$ (where w_i is the normalized weight of particle i) falls below a threshold $\frac{n}{r}$, $r > 1$, a new set of particles is generated using a Monte-Carlo Markov Chain algorithm. On the newly generated samples, the importance weight (before normalization) of iterations $k' > k$ is $\frac{p(w;\, \mathcal{U}'_{k'})}{p(w;\, \mathcal{U}_k)} = p(w;\, \mathcal{U}_{k,k'})$, where the archive $\mathcal{U}_{k,k'}$ only contains constraints of time-stamp $i, k < i \leq k'$. IBPS iterates the importance sampling-based selection, using MCMC to generate new particles whenever the number of number of effective particles falls below the threshold. For the MCMC algorithm, the proposal distribution $q(\mathbf{w}' \mid \mathbf{w})$ is defined as follows: a radius d is chosen uniformly on $[d_{min}, d_{max}]$ and a new point uniformly sampled from the ball $\mathcal{B}(\mathbf{w}, d)$ is projected on $\mathbf{W}$ by normalization. Since distribution q only depends on the distance between $\mathbf{w}'$ and $\mathbf{w}$ it follows that $q(\mathbf{w}' \mid \mathbf{w}) = q(\mathbf{w} \mid \mathbf{w}')$. Hence their ratio cancels out in the MCMC algorithm and $q(\mathbf{w}' \mid \mathbf{w})$ does not need to be evaluated.

2.5 IBPS Algorithm

The IBPS algorithm is as follows: i) A first random trajectory is generated; ii) The agent evaluates (section 2.4) and optimizes (section 2.3) the EUS_{NL} criterion, and returns the policy π with best empirical average EUS_{NL} over $\mathbf{u}$ sampled from π; iii) One trajectory is sampled from π; iv) The expert qualifies this trajectory as "better" or "worse" than the former best trajectory; v) The posterior $p(w; \mathcal{U}_t)$ is updated (section 2.1) and the process is iterated from ii). At any given time, if the agent is asked to stop learning and return a policy, he returns the one maximizing the EU, i.e. $\pi = \arg\max_\pi EU(\mathbb{E}_{u\sim\pi}(\mathbf{u}), \mathcal{U}_t)$.

3 Experimental results

This section presents two experimental validations of the IBPS approach. The first setting considers the in-situ interaction between a robot and a human expert, and demonstrates the applicability of IBPS to real world problems. The

[5] Note that while the used RL algorithms assume infinite length trajectories, only finite-length trajectories will be demonstrated to the expert. The trajectory length H must thus be adjusted depending on the discount factor γ, to ensure that the cumulative reward for $t > H$ can be safely discarded.

second setting considers a simulated grid world problem and presents a sensibility analysis of both the agent and the simulated expert noise models, and their interaction.

3.1 Reaching a target robot

The first problem, inspired from the swarm robotics framework [21], aims at having a swarm robot aligned in a very precise way with another robot to dock to each other and form a multi-robot organism. The simplified setting considered here involves a single e-puck robot equipped with a (52x39, 4img/s) camera; the target behavior is to reach an immobile robot with its leds turned on. The starting state, from the perspective of the expert is displayed in Fig. 1.a. Sixteen states and five macro-actions are manually defined; The macro-actions involve: stay motionless for one time step, moving forward (resp. backward) for 3 time steps, and rotate to the left (resp. to the right) for one time step. The mapping from the robot camera image onto the state space $\{1, \dots 16\}$ is defined as follows. Given the set of pixels $S_{lum} = \{(x_i, y_i)\}$ associated to the target (the stationary robot) leds, the distance to the target is defined by discretizing $max(y_i)$ in 5 values; the orientation w.r.t. the target is defined by discretizing $\frac{1}{2}(min(x_i) + max(x_i))$ in 3 values. The case where S_{lum} is the empty set (no target in sight) corresponds to the 16th state.

The parametric policy space is finite, associating one action to each state. The parametric-to-behavioral space is defined as follows. Letting the current trajectory generated from policy $\pi_{\mathbf{x}}$ be noted $s_0, \dots s_{H-1}$, where $s_i \in \{1, \dots, 16\}$ is the index of the robot state at time i, the associated behavioral representation is set to $\Phi(\mathbf{x}) \in \mathbb{R}^{16}$ with $\Phi(\mathbf{x})[j] = \sum_{h=0}^{H-1} \gamma^h \delta_{j,s_h}$ where δ_{j,s_h} is 1 iff the robot is in state j at time h, and 0 otherwise. Parameter γ is set to .95 in the experiment. The resulting $\mathbf{u}$ was normalized thereafter s.t. $||\mathbf{u}||_1 = 1$. M was set to 1/2 in the response model. The robot demonstrates the selected policy for 80 time steps; it then gets the expert's feedback and the expert sets the robot back to its initial position. The feedback is interpreted using a built-in procedure: the expert activates the front (resp.) the back e-puck sensors to indicate that the current behavior is better (resp. worse) than the previous best one. Due to the small scale of the experiment and the fast convergence, MCMC was not necessary and the integration on $\mathbf{W}$ was estimated using importance sampling from an initial set of $50,000$ particles drawn uniformly from the hypersphere in $\mathbb{R}^{16}$. During the EUS_{NL} optimization phase, 50 initial utilities $\mathbf{w}_0$ sampled from $p(w, \mathcal{U}_t)$ were considered and a Policy Iteration algorithm was used, taking as input a provided transition matrix and $\mathbf{w}$ and returning the average trajectory $\bar{\mathbf{u}}$ of the policy maximizing $\mathbf{w}$.

Figure 1.b shows the visit to the goal state (averaged out of 5 runs) *vs* the number of expert's feedback, a.k.a. number of interactions. Interestingly, all 5 policies were found to reach the goal state after 5 interactions; the observed performance decrease is explained by inspecting the logs, showing that the expert made an error in one of the runs, favoring a trajectory that spent significantly less time in the goal state. Figs. 1.c shows the average utility weight vector

vs the number of interactions for this particular run. Interestingly, the weight associated to some states increases after the first interactions, and thereafter decreases, due to the fact that these states are intermediate between the starting and the goal states. Fig. 1.c is the run where the expert made a mistake (at the third feedback), causing the weight associated to the "nothing in sight" state to increase and out-pass the other weights; this in turn leads astray the optimal policy; however IBPS recovers one iteration later. Concerning the execution time, since the RL is based on the transition matrix, almost all of the time is solely consumed by the demonstrations of trajectories to the expert.

3.2 Reaching a target state in a grid world

The second problem (Fig. 1.d) is concerned with reaching a goal state in a simulated grid world, where the state space includes 25 states and the action space includes 5 actions (up, down, left, right or stay motionless). The robot stays motionless with probability 1/2 upon selecting the action up, down, left or right (respectively with probability 1 if the selected action would send it in the wall). The expert's feedback is emulated according to the true utility $\mathbf{w}^*$ shown in Fig. 1.d (up to renormalization) and a noisy response model with hyper-parameter M_E; M_A is the hyper-parameter of the expert noise model estimated by the agent, with M_E and M_A ranging in $\{1, 1/2, 1/4\}$ subject to $M_A \geq M_E$). A large M_E is interpreted as less competent expert; a large M_A means that the agent underestimates the expert's competence, resulting in slow updates of the posterior. Time horizon is set to $H = 300$ and discount γ to .95. Same parameters as in section 3.1 are used for the optimization of EUS_{NL}, and the integration is estimated using $n = 10,000$ particles, re-sampled each time the number of effective particles falls under $n/5$ (section 2.4). During re-sampling, the particle with highest weight is selected as the initial point for MCMC and n new particles are selected from a chain of length $500n$ (the rest of the particles being discarded). For the proposal distribution, parameters d_{min} and d_{max} are respectively set to .4 and 3.

Fig. 1.e shows the true utility of the optimal policy w.r.t. the expected utility, averaged out of 21 independent runs, vs the number of expert's feedbacks for all six (M_E, M_A) settings. Fig. 1.e indicates that the agent learning speed primarily depends on M_E, and secondarily on M_A (as $(M_E = 1/4, M_A = 1)$ converges faster than $(M_E = 1/2, M_A = 1/2)$) However Fig. 1.f shows that the combined impact of M_E and M_A is more intricate than thought at first sight. Indeed, by computing the frequency of expert mistakes during the first k interactions $(k = 1, 60)$, it is seen that the expert makes more mistakes when $M_A = 1$ than when $M_A = 1/4$ for same $M_E = 1/4$. This is explained by the fact that the error rate does not only depend on the expert's competence but also on the relevance of the comparisons he is provided with. For $M_A = 1/4$, the agent learns faster, thus submitting more relevant queries to the expert, thus priming a virtuous educational process. This also explains the fast decrease of the error rate for $M_E = 1/4, M_A = 1/4$ which seems to follow a different regime (empirically faster than linear) than $M_E = 1/4, M_A = 1$. Fig 1.f is also in line with the

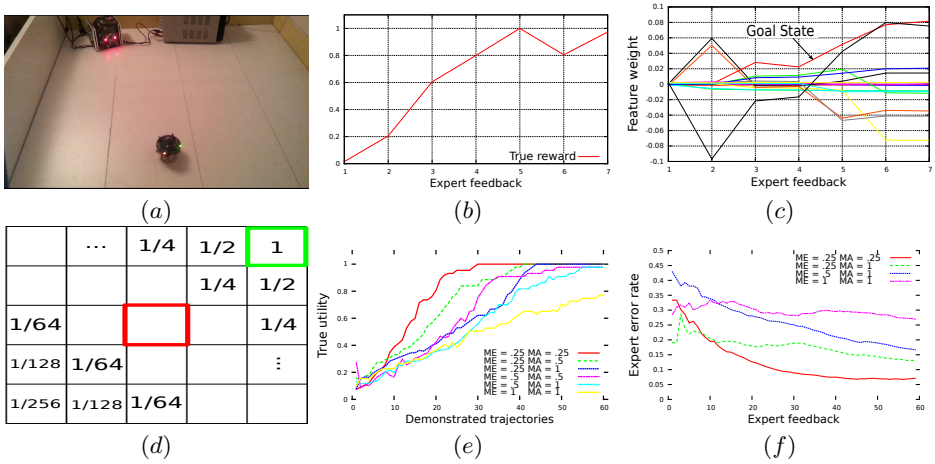

Fig. 1. a) Initial state from an observer's perspective. b) Performance of the policy maximizing the average utility *vs* number of interactions with the expert, averaged out of 5 runs. c) Average utility weight vector in $\mathbb{R}^{16}$ *vs* number of interactions for a representative run. d) The grid world: the agent initially in the center state must visit the upper rightmost goal state. The numbers represent the true hidden utility; e) Performance of the policy maximizing the average utility *vs* number of interactions with the expert, averaged out of 21 runs. f) Expert error rate up *vs* number of interactions.

intuition that the most delicate stage of the algorithm is the initial one, where the demonstrated trajectories are of low utility, making them harder to compare and increasing the probability of expert mistakes.

4 Discussion and Perspectives

The main contribution of the present paper is to show the feasibility of training a robot on-board, online, along an active preference-based learning framework, by interacting with *a fallible human expert*. The Bayesian IBPS framework enables the robot to gradually model both the expert's preferences and the expert's competence.

Several lessons are learned from the preliminary experiments in-situ and in simulation. Unsurprisingly, the human experts (the authors) do make errors, leading astray the learning robot; but IBPS is shown to recover from such errors in the simple experimental settings considered. The main lesson regards the intricate interaction between the learning robot and the human expert: an pessimistic competence model leads the robot to present the expert with poorly informative queries, *thereby increasing the probability for the expert to make errors* and be deemed inconsistent. A cumulative (dis)advantage phenomenon is thus observed: a hyper-linear progress (decrease of expert's errors and increase of agent skills) are observed when the agent trusts a competent expert. The im-

portance of a good educational start is witnessed as poor initial demonstrations lead to a poor utility model, leading itself to poorly informative queries.

Further work, inspired from [24], will investigate how to extend IBPS in order to learn effectively from short and well focused sub-behaviors. Indeed, the better framed the sub-behaviors, the more informative the expert's preferences will be. The challenge is to identify the most relevant starting states, conditioning the behavior segmentation. A multiple instance ranking approach [4] is envisioned, considering a long trajectory as a set of sub-behaviors where a few sub-behaviors are responsible for the expert's preferences. Incidentally this approach would alleviate the IBPS dependency on the starting state.

Bibliography

[1] Abbeel, P., Ng, A.: Apprenticeship learning via inverse reinforcement learning. In: Brodley, C.E. (ed.) ICML. ACM Int. Conf. Proc. Series, vol. 69. ACM (2004)

[2] Akrour, R., Schoenauer, M., Sebag, M.: Preference-based policy learning. In: ECML/PKDD (1). pp. 12–27. No. 6911, Springer Verlag LNCS (2011)

[3] Akrour, R., Schoenauer, M., Sebag, M.: April: Active preference learning-based reinforcement learning. In: ECML/PKDD (2). vol. 7524, pp. 116–131. Springer LNCS (2012)

[4] Bergeron, C., Zaretzki, J., Breneman, C.M., Bennett, K.P.: Multiple instance ranking. In: ICML. pp. 48–55 (2008)

[5] Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific, 1st edn. (1996)

[6] Calinon, S., Guenter, F., Billard, A.: On Learning, Representing and Generalizing a Task in a Humanoid Robot. IEEE Trans. on systems, man and cybernetics 37(2), 286–298 (2007)

[7] Cheng, W., Fürnkranz, J., Hüllermeier, E., Park, S.H.: Preference-based policy iteration: Leveraging preference learning for reinforcement learning. In: ECML/PKDD (1). pp. 312–327. No. 6911, Springer Verlag LNCS (2011)

[8] Chu, W., Ghahramani, Z.: Preference learning with gaussian processes. In: ICML. pp. 137–144 (2005)

[9] Dimitrakakis, C., Lagoudakis, M.G.: Rollout sampling approximate policy iteration. Machine Learning 72(3), 157–171 (2008)

[10] Heidrich-Meisner, V., Igel, C.: Hoeffding and bernstein races for selecting policies in evolutionary direct policy search. In: ICML. p. 51 (2009)

[11] Klein, E., Geist, M., Piot, B., Pietquin, O.: Inverse reinforcement learning through structured classification. In: NIPS. pp. 1016–1024 (2012)

[12] Kober, J., Peters, J.: Reinforcement Learning in Robotics: A Survey, pp. 579–610. Springer Berlin Heidelberg (2012)

[13] Kolter, J.Z., Abbeel, P., Ng, A.Y.: Hierarchical apprenticeship learning with application to quadruped locomotion. In: NIPS. MIT Press (2007)

[14] Konidaris, G., Kuindersma, S., Barto, A., Grupen, R.: Constructing skill trees for reinforcement learning agents from demonstration trajectories. In: NIPS 23. pp. 1162–1170 (2010)

[15] Lagoudakis, M., Parr, R.: Least-squares policy iteration. Journal of Machine Learning Research (JMLR) 4, 1107–1149 (2003)

[16] Luce, R.D.: Individual choice behavior. John Wiley, New York (1959)

[17] Ng, A., Russell, S.: Algorithms for inverse reinforcement learning. In: Langley, P. (ed.) ICML. pp. 663–670. Morgan Kaufmann (2000)

[18] Shepard, R.N.: Stimulus and response generalization: A stochastic model relating generalization to distance in psychological space. Psychometrika 22, 325345 (1957)

[19] Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
[20] Sutton, R.S., Maei, H.R., Precup, D., Bhatnagar, S., Silver, D., Szepesvri, C., E, W.: Fast gradient-descent methods for temporal-difference learning with linear function approximation. In: ICML (2009)
[21] Symbrion: FP7 European Project FET IP 216342 (Jun 2009), http://symbrion.eu/
[22] Szepesvári, C.: Algorithms for Reinforcement Learning. Morgan & Claypool (2010)
[23] Viappiani, P., Boutilier, C.: Optimal Bayesian recommendation sets and myopically optimal choice query sets. In: NIPS. pp. 2352–2360 (2010)
[24] Wilson, A., Fern, A., Tadepalli, P.: A bayesian approach for policy learning from trajectory preference queries. In: NIPS. pp. 1142–1150 (2012)

5 Supplementary Material

This supplementary material contains the proof of Lemma 1 and Theorem 1, in addition to proofs of two claims made in Section 2.3.

5.1 Proof of the Lemma 1

Lemma 1 states that for any set $q = \{\mathbf{u}, \mathbf{u}'\}$ of two trajectories $\mathbf{u}$ and $\mathbf{u}'$, and for any archive $\mathcal{U}_t$, we have:

$$\mathrm{EUS}_{NL}(q,\mathcal{U}_t) - \frac{M}{2\lambda}(1 - \frac{1+ln\lambda}{\lambda}) \leq \mathrm{EUS}_N(q;\mathcal{U}_t) \leq \mathrm{EUS}_{NL}(q,\mathcal{U}_t)$$

With $\lambda = e^{-\frac{1}{2}-W_{-1}(-\frac{1}{2}e^{-\frac{1}{2}})}$ and W_{-1} is the lower branch of the Lambert W function. Let $L_N = \frac{M}{2\lambda}(1 - \frac{1+ln\lambda}{\lambda}) \approx \frac{M}{19.6433}$

Proof Let $q = \{\mathbf{u}, \mathbf{u}'\}$ and $\mathbf{z}_w = \mathbf{w}(\mathbf{u} - \mathbf{u}')$. By definition of $\mathrm{EUS}_{NL}(q,\mathcal{U}_t)$ and $\mathrm{EUS}_N(q;\mathcal{U}_t)$ and since $P_N(-\mathbf{z}_w) = 1 - P_N(\mathbf{z}_w)$, we have:

$$\begin{aligned}
\mathrm{EUS}_{NL}(\{\mathbf{u}, \mathbf{u}'\},\mathcal{U}_t) - \mathrm{EUS}_N(\{\mathbf{u}, \mathbf{u}'\};\mathcal{U}_t) &= \int_W \max(\mathbf{wu}, \mathbf{wu}') - \mathbf{wu}P_N(\mathbf{z}_w) - \mathbf{wu}'P_N(-\mathbf{z}_w)p(\mathbf{w};\mathcal{U}_t)\mathrm{d}\mathbf{w} \\
&= \int_W \max(\mathbf{wu}, \mathbf{wu}') - \mathbf{wu}' - \mathbf{z}_w P_N(\mathbf{z}_w)\mathrm{d}\mathbf{w} \\
&\stackrel{\text{def}}{=} \int_W \Delta(\mathbf{z}_w)p(\mathbf{w};\mathcal{U}_t)\mathrm{d}\mathbf{w}
\end{aligned}$$

It is clear from this equations that EUS_{NL} can never be smaller than EUS_N since for any $\mathbf{w}$, no convex combination of $\mathbf{wu}$ and $\mathbf{wu}'$ can exceed their maximum. This also implies that $\Delta(\mathbf{z}_w) \geq 0$ for all $\mathbf{w} \in \mathbf{W}$. We want now an upper bound of $\Delta(\mathbf{z}_w)$. For any $\mathbf{w}$ s.t. $\mathbf{z}_w \geq 0$, $\Delta(\mathbf{z}_w) = \mathbf{z}_w[1 - P_N(\mathbf{z}_w)]$ and for $\mathbf{z}_w \leq 0, \Delta(\mathbf{z}_w) = -\mathbf{z}_w[1 - P_N(-\mathbf{z}_w)]$. It then suffices to upper bound $\mathbf{z}_w[1 - P_N(\mathbf{z}_w)]$ for $\mathbf{z}_w \geq 0$. If $\mathbf{z}_w \geq M$ or $\mathbf{z}_w = 0$, we have $\Delta(\mathbf{z}_w) = 0$. And for $0 < z < M$ we search:

$$\max_{z_w} \Delta(\mathbf{z}_w) = \max_{z_w} \frac{1}{2}(\mathbf{z}_w - {\mathbf{z}_w}^2(\frac{1+lnM}{M}) + \frac{{\mathbf{z}_w}^2 ln\mathbf{z}_w}{M})$$

We compute the derivative w.r.t. $\mathbf{z}_w$:

$$\Delta'(\mathbf{z}_w) = \frac{1}{2}(1 - \mathbf{z}_w(\frac{1+2lnM}{M}) + \frac{2\mathbf{z}_w ln\mathbf{z}_w}{M})$$

And use the variable substitution $\mathbf{z}_w = Me^{t+\frac{1}{2}}, -\infty < t < -\frac{1}{2}$ to solve the equation:

$$\Delta'(\mathbf{z}_w) = 0 \Leftrightarrow \Delta'(Me^{t+\frac{1}{2}}) = 0 \Leftrightarrow te^t = -\frac{1}{2}e^{-\frac{1}{2}}$$

This equation accepts an obvious solution $t_1 = -\frac{1}{2}$ that is rejected as $t < -\frac{1}{2}$ (it corresponds to $\mathbf{z}_w = M$) and a second solution from the lower branch of the Lambert W function W_{-1}, $t_2 = W_{-1}(-\frac{1}{2}e^{-\frac{1}{2}}) \approx -1.7564$. These are the only two values that nullify $\Delta'(\mathbf{z}_w)$. Let $\lambda = e^{t_2+\frac{1}{2}}$. The maximum of Δ is reached at $\Delta(\frac{M}{\lambda}) = \frac{M}{2\lambda}(1 - \frac{1+ln\lambda}{\lambda})$.
This bound holds for any $\mathbf{w}$ and thus will hold after integrating over $\mathbf{W}$ for any archive $\mathcal{U}_t$. □

5.2 Proof of the Theorem 1

Theorem 1 states that $\mathrm{EUS}^*_N(\mathcal{U}_t) \geq \mathrm{EPU}^*_N(\mathcal{U}_t) - L_N$

Proof The consequence of Lemma 2 in [23] is that $\mathrm{EPU}^*_{NL}(\mathcal{U}_t) \geq \mathrm{EPU}^*_N(\mathcal{U}_t)$. This holds for any noise model since it only depends on the fact that no convex combination of a set of utilities can exceed their maximum. and since $\mathrm{EUS}^*_{NL}(\mathcal{U}_t) = \mathrm{EPU}^*_{NL}(\mathcal{U}_t)$. It thus follows, using this two results and Lemme 1 that $\mathrm{EPU}^*_N(\mathcal{U}_t) \leq \mathrm{EPU}^*_{NL}(\mathcal{U}_t) = \mathrm{EUS}^*_{NL}(\mathcal{U}_t) \leq \mathrm{EUS}^*_L(\mathcal{U}_t) + L_N$. □

5.3 Proof of claims made in Section 2.3

Claim 1 In the optimization part of EUS_{NL}, we stated $\mathrm{EUS}_{NL}(\bar{\mathbf{u}}_0; \mathbf{u}_t, \mathcal{U}_t) \leq \mathrm{EUS}_{NL}(\bar{\mathbf{u}}_1; \mathbf{u}_t, \mathcal{U}_t) \leq \cdots \leq \mathrm{EUS}_{NL}(\bar{\mathbf{u}}_{k+1}; \mathbf{u}_t, \mathcal{U}_t)$. To prove it we first decompose EUS_{NL}:

$$\begin{aligned}
\mathrm{EUS}_{NL}(\bar{\mathbf{u}}_k; \mathbf{u}_t, \mathcal{U}_t) &= \int_{\mathbf{W}} \max(\mathbf{w}\bar{\mathbf{u}}_k, \mathbf{w}\mathbf{u}_t)\ p(\mathbf{w}; \mathcal{U}_t)\ \mathrm{d}\mathbf{w} \\
&= \int_{\mathbf{W}|(\mathbf{w}\bar{\mathbf{u}}_k > \mathbf{w}\mathbf{u}_t)} \mathbf{w}\bar{\mathbf{u}}_k p(\mathbf{w}; \mathcal{U}_t)\mathrm{d}\mathbf{w} + \int_{\mathbf{W}|(\mathbf{w}\bar{\mathbf{u}}_k \leq \mathbf{w}\mathbf{u}_t)} \mathbf{w}\mathbf{u}_t p(\mathbf{w}; \mathcal{U}_t)\mathrm{d}\mathbf{w}
\end{aligned}$$

Since $\bar{u}_{k+1}$ is obtained by maximizing $\mathbf{w}_k = \int_W \mathbf{w}\ p(\mathbf{w} \mid \mathcal{U}_t) 1_{\{w \cdot \bar{u_k} > w \cdot u_t\}}\ \mathrm{d}\mathbf{w}$ then $\bar{u}_{k+1}$ will perform better on the subpart of $\mathbf{W}$ where $\mathbf{W} \mid (\mathbf{w}\bar{\mathbf{u}}_k > \mathbf{w}\mathbf{u}_t)$. It follows that:

$$\begin{aligned}
\mathrm{EUS}_{NL}(\bar{\mathbf{u}}_k; \mathbf{u}_t, \mathcal{U}_t) &\leq \int_{\mathbf{W}|(\mathbf{w}\bar{\mathbf{u}}_k > \mathbf{w}\mathbf{u}_t)} \mathbf{w}\bar{\mathbf{u}}_{k+1} p(\mathbf{w}; \mathcal{U}_t)\mathrm{d}\mathbf{w} + \int_{\mathbf{W}|(\mathbf{w}\bar{\mathbf{u}}_k \leq \mathbf{w}\mathbf{u}_t)} \mathbf{w}\mathbf{u}_t p(\mathbf{w}; \mathcal{U}_t)\mathrm{d}\mathbf{w} \\
&\leq \int_{\mathbf{W}} \max(\mathbf{w}\bar{\mathbf{u}}_{k+1}, \mathbf{w}\mathbf{u}_t)\ p(\mathbf{w}; \mathcal{U}_t)\ \mathrm{d}\mathbf{w} \\
&\stackrel{\text{def}}{=} \mathrm{EUS}_{NL}(\bar{\mathbf{u}}_{k+1}; \mathbf{u}_t, \mathcal{U}_t)
\end{aligned}$$

The second inequality is explained by decomposing the integrals on each $\mathbf{w}$: for each $\mathbf{w} \in (\mathbf{W} \mid (\mathbf{w}\bar{\mathbf{u}}_k > \mathbf{w}\mathbf{u}_t))$, if $\mathbf{w}\mathbf{u}_{k+1} > \mathbf{w}\mathbf{u}_t$ then in both the left and right sides of the inequalities we associate to $\mathbf{w}$, $\mathbf{w}\mathbf{u}_{k+1}$. However in the case where $\mathbf{w}\mathbf{u}_{k+1} < \mathbf{w}\mathbf{u}_t$, in the left side of the inequality $\mathbf{w}\mathbf{u}_{k+1}$ is associated to $\mathbf{w}$ while in the right side $\mathbf{w}\mathbf{u}_t$ is associated to $\mathbf{w}$, which will increase the expectation since $\mathbf{w}\mathbf{u}_{k+1} < \mathbf{w}\mathbf{u}_t$. And similarly for $\mathbf{w} \in (\mathbf{W} \mid (\mathbf{w}\bar{\mathbf{u}}_k \leq \mathbf{w}\mathbf{u}_t))$, we see that the term associated to each $\mathbf{w}$ can only increase in the right side of the inequality.□

Claim 2 We claim that for any feature function ϕ and utility $\mathbf{w}$, we can define a feature function ϕ' with $\phi'_j(s^i) = 1$ if $i = j$, and 0 otherwise and utility $\mathbf{w}'$ s.t. $\mathbf{w}'_i = \mathbf{w} \cdot \phi(s^i)$ having the same optimal policy as with ϕ and $\mathbf{w}$. If we let the matrix Φ, were the i^{th} column of $\Phi, \Phi_i = \phi(s^i)$, $\mathbf{w}$ and $\mathbf{w}'$ row vectors, and ϕ and ϕ' column vectors. Since in this case $\Phi \cdot \phi'(s^i) = \phi(s^i)$ then:

$$\begin{aligned}
\mathbf{w}' \sum_{t=0}^{t=\infty} \phi'(s^t) &= \mathbf{w} \cdot \Phi \sum_{t=0}^{t=\infty} \phi'(s^t) \\
&= \mathbf{w} \sum_{t=0}^{t=\infty} \Phi \cdot \phi'(s^t) \\
&= \mathbf{w} \sum_{t=0}^{t=\infty} \phi(s^t)
\end{aligned}$$

And for a trajectory $\mathbf{u}' = \sum_{t=0}^{t=\infty} \gamma^t \phi'(s_t)$ we can get back to a trajectory $\mathbf{u} = \sum_{t=0}^{t=\infty} \gamma^t \phi(s^t)$ without knowing the sequence of visited states $s_0 \dots s_\infty$ using the transformation:

$$\begin{aligned}
\mathbf{u} &= \sum_i \mathbf{u}'_i \phi(s^i) \\
&= \sum_i [\sum_{t=0}^{t=\infty} \gamma^t \phi'(s_t)]_i \phi(s^i) \\
&= \sum_{t=0}^{t=\infty} \gamma^t \sum_i [\phi'(s_t)]_i \phi(s^i) \\
&= \sum_{t=0}^{t=\infty} \gamma^t \phi(s^t), \text{ since } [\phi'(s_t)]_i = 1 \text{ iff } i = t, \text{ and 0 else}
\end{aligned}$$

□