Foundations and Trends® in Robotics

Interactive Imitation Learning in Robotics: A Survey

Suggested Citation: Carlos Celemin, Rodrigo Pérez-Dattari, Eugenio Chisari, Giovanni Franzese, Leandro de Souza Rosa, Ravi Prakash, Zlatan Ajanović, Marta Ferraz, Abhinav Valada and Jens Kober (2022), "Interactive Imitation Learning in Robotics: A Survey", Foundations and Trends® in Robotics: Vol. 10, No. 1-2, pp 1–197. DOI: 10.1561/2300000072.

Carlos Celemin
Delft University of Technology
c.e.celeminpaez@tudelft.nl

Rodrigo Pérez-Dattari
Delft University of Technology
r.j.perezdattari@tudelft.nl

Eugenio Chisari
University of Freiburg
chisari@cs.uni-freiburg.de

Giovanni Franzese
Delft University of Technology
g.franzese@tudelft.nl

Leandro de Souza Rosa
Delft University of Technology
l.desouzarosa@tudelft.nl

Ravi Prakash
Delft University of Technology
r.prakash-1@tudelft.nl

Zlatan Ajanović
Delft University of Technology
z.ajanovic@tudelft.nl

Marta Ferraz
Delft University of Technology
m.ferraz@tudelft.nl

Abhinav Valada
University of Freiburg
valada@cs.uni-freiburg.de

Jens Kober
Delft University of Technology
j.kober@tudelft.nl

now
the essence of knowledge
Boston — Delft

Contents

1 Introduction **3**
- 1.1 Motivation . . . 3
- 1.2 Terminology Unification . . . 6
- 1.3 Others Surveys and Outline . . . 11

2 Theoretical Background **15**
- 2.1 Decision Theory . . . 15
- 2.2 Interactive Imitation Learning . . . 18

3 Modalities of Interaction **26**
- 3.1 Human Feedback in Evaluative Space . . . 28
- 3.2 Human Feedback in Transition (State-Action) Space . . . 38
- 3.3 Discussion . . . 50

4 Behavior Representations Learned from Interactions **55**
- 4.1 Direct Policy Learning (Actions) . . . 56
- 4.2 Learning Desired State Transition/Dynamics . . . 58
- 4.3 Learning Reward and Objective Functions . . . 60
- 4.4 Discussion . . . 63

5 Auxiliary Models **65**
- 5.1 Task Features Learning . . . 65
- 5.2 Object Affordances . . . 68

5.3 Forward and Inverse Transition Models 70
5.4 Confidence, Novelty and Risk Models 71
5.5 Human Models for Feedback Interpretation 73
5.6 Discussion 74

6 Model Representations (Function Approximation) 76
6.1 Linear Models 77
6.2 Gaussian Process 79
6.3 Gaussian Mixture Model 80
6.4 Support Vector Machine 80
6.5 Neural Networks 80
6.6 Movement-Conditioned Models 82
6.7 Discussion 84

7 On/Off Policy Learning 85
7.1 Online and Offline Learning 86
7.2 On-policy and Off-policy Learning 87
7.3 On-Policy/Off-Policy Learning in Imitation Learning 92
7.4 Discussion 96

8 Reinforcement Learning with Human-in-the-Loop 99
8.1 Other Related Approaches 100
8.2 Historical Perspective 100
8.3 Reinforcement Learning with Human-in-the-Loop Approaches 102
8.4 Discussion 107

9 Interfaces 108
9.1 Human-to-Robot Interfaces 109
9.2 Robot-to-Human Interfaces 115
9.3 Interface Design 117
9.4 Discussion 117

10 User Studies in IIL 119
10.1 Study Setup 119
10.2 Evaluation Methods 123
10.3 Discussion 129

11 Benchmarks and Applications **130**
11.1 Applications 131
11.2 Datasets 144
11.3 Benchmarks 145
11.4 Discussion 148

12 Research Challenges and Opportunities **150**

13 Conclusion **155**

Author Contributions **157**

Glossary **159**

References **165**

Interactive Imitation Learning in Robotics: A Survey

Carlos Celemin[1*], Rodrigo Pérez-Dattari[1*], Eugenio Chisari[2*], Giovanni Franzese[1*], Leandro de Souza Rosa[1], Ravi Prakash[1], Zlatan Ajanović[1], Marta Ferraz[1], Abhinav Valada[2] and Jens Kober[1]

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

[2] *Robot Learning Lab, University of Freiburg, Germany*

ABSTRACT

Interactive Imitation Learning (IIL) is a branch of Imitation Learning (IL) where human feedback is provided intermittently during robot execution allowing an online improvement of the robot's behavior.

In recent years, IIL has increasingly started to carve out its own space as a promising data-driven alternative for solving complex robotic tasks. The advantages of IIL are twofold, 1) it is data-efficient, as the human feedback guides the robot directly towards an improved behavior (in contrast with Reinforcement Learning (RL), where behaviors must be discovered by trial and error), and 2) it is robust, as the distribution mismatch between the teacher and learner trajectories is minimized by providing feedback directly over the learner's trajectories (as opposed to offline IL methods such as Behavioral Cloning).

Nevertheless, despite the opportunities that IIL presents, its terminology, structure, and applicability are not clear

*These authors contributed equally to this work.

Carlos Celemin, Rodrigo Pérez-Dattari, Eugenio Chisari, Giovanni Franzese, Leandro de Souza Rosa, Ravi Prakash, Zlatan Ajanović, Marta Ferraz, Abhinav Valada and Jens Kober (2022), "Interactive Imitation Learning in Robotics: A Survey", Foundations and Trends® in Robotics: Vol. 10, No. 1-2, pp 1–197. DOI: 10.1561/2300000072.

nor unified in the literature, slowing down its development and, therefore, the research of innovative formulations and discoveries.

In this work, we attempt to facilitate research in IIL and lower entry barriers for new practitioners by providing a survey of the field that unifies and structures it. In addition, we aim to raise awareness of its potential, what has been accomplished and what are still open research questions.

We organize the most relevant works in IIL in terms of human-robot interaction (i.e., types of feedback), interfaces (i.e., means of providing feedback), learning (i.e., models learned from feedback and function approximators), user experience (i.e., human perception about the learning process), applications, and benchmarks. Furthermore, we analyze similarities and differences between IIL and RL, providing a discussion on how the concepts *offline*, *online*, *off-policy* and *on-policy* learning should be transferred to IIL from the RL literature.

We particularly focus on robotic applications in the real world and discuss their implications, limitations, and promising future areas of research.

1

Introduction

1.1 Motivation

Existing robotic technology is still mostly limited to being used by expert programmers who can adapt the systems to new required conditions, but not flexible and adaptable by non-expert workers or end-users. Imitation Learning (IL) has obtained considerable attention as a potential direction for enabling all kinds of users to easily program the behavior of robots or virtual agents. The teaching process takes place directly in the application context, in a natural way for humans, and does not require engineering effort to adapt the behavior for each different scenario.

In the case teachers (i.e., humans with knowledge about the task) are available and able to transfer their knowledge to the agent, it is preferred to program behaviors from recorded demonstrations rather than tackling the problem with other Machine Learning (ML) techniques such as Reinforcement Learning (RL), which involve additional design, infrastructure, safety, and data efficiency challenges (Sutton and Barto, 2018), and in many cases are not applicable to physical systems due to time and resource limitations.

When considering the advantages of programming robots in a natural way, like we humans do for teaching complex skills (e.g., requiring fast dynamics and dexterity) to others, the possibilities are not limited to recording demonstrations, for later fitting a policy model, as it is done in traditional IL methods (Argall *et al.*, 2009). In practice, an initial set of demonstrations or instructions tend to suffice to teach very simple and easy tasks from human to human, e.g., the instructions for opening a door, plugging a phone charger, or the user guide for most devices we use on a daily basis. Nevertheless, for complex skills such as playing a sport, a loop of interactions is required for learning, because then the teacher explains/shows the student what to do by directly correcting/evaluating its actions, improving its behavior over past mistakes and successes. Otherwise, considering and explaining all possible scenarios in advance would be intractable for both the teacher and the student.

This kind of teaching is based on different types of teaching feedback, like demonstrations, sporadic corrections, or evaluations (grading) with value judgments or rankings. As an example, when teaching a complex skill like playing tennis, various steps can be involved. The teacher shows full demonstrations of the stroke themselves to the learner. When the student tries to replicate the example, the teacher can show what a better execution would look like. After the student performs the stroke, the teacher could advise with voice instructions to slightly correct the angles, velocities, or forces of the movement. Moreover, the teacher can sporadically congratulate the student or make it clear that some decisions were not so good. This kind of interactive teaching approach seems to be, for humans, the most natural strategy for teaching to perform more complex skills; therefore, it is desirable to teach robots in the same way.

In recent years, the domains of robotics and ML have increasingly adopted and developed these interactive teaching strategies, as can be observed in Figure 1.1. In this work, Interactive Imitation Learning (IIL) refers to all the methods that include the teacher in the learning loop for training sequential decision-making systems. The objective of this work is to survey the literature on these methods and to present the most relevant observations in an organized structure.

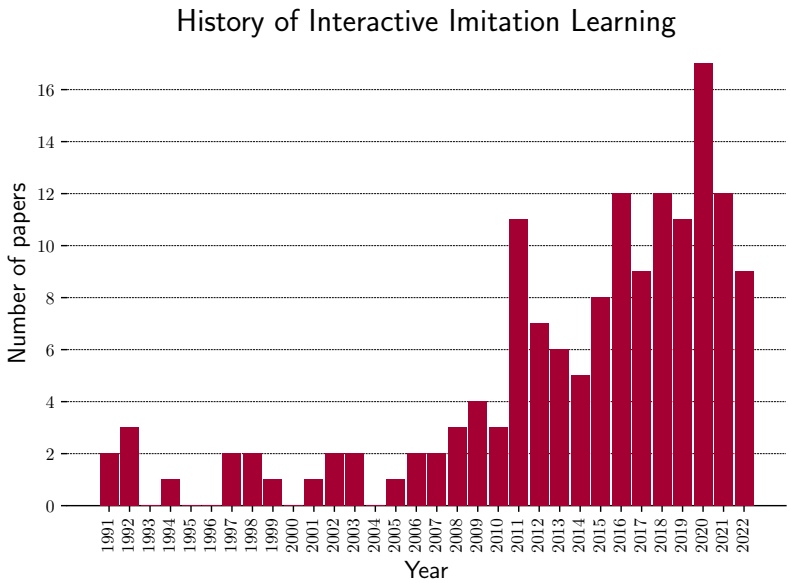


Figure 1.1: Histogram of IIL papers (from the group of works surveyed until the beginning of 2022) written per year.

The study of IIL methods has increased and the community has grown because these strategies introduce additional benefits with respect to learning paradigms such as traditional IL. Some of those advantages are:

- A more natural or intuitive teaching approach.
- Enabling users who are non-experts at demonstrating the task to teach successful policies.
- Obtaining richer datasets consisting of data from situations that are not faced when learning from full demonstrations, as the distribution of data collected is induced by the learner instead of the teacher, avoiding data mismatch issues (see Section 2.2.6).
- More flexibility to the teachers, who are not constrained to use only demonstrations for transferring their knowledge, but they

can use other kinds of feedback, like relative corrections, human reinforcements, or comparisons.

- Offers alternatives to solve the correspondence problem that exists between the space where teachers can give demonstrations and the space where the robot executes the actions.

- Some methods have more tolerance for the teacher's mistakes or provide a better approach to compensate for them.

Nonetheless, there are certain challenges that should be considered when a teacher is in the learning loop. Human teachers can be inconsistent and make mistakes, there is uncertainty in their input that tries to explain their intention, they need to learn to adapt to the changing behavior of the learning agent, and the learning process is open-ended (Dudley and Kristensson, 2018).

In this work, we review the context that defines the domain of IIL and how it relates to other known learning approaches. We highlight the most relevant aspects to be considered for teaching an agent interactively and organize the methods according to them. This study is based on grouping and surveying the most relevant established papers in the literature, along with more recent follow-up works that have shown promising contributions. All these papers were gathered in a set of works used as reference for organizing the different classifications proposed throughout the different sections. This set is also used for generating the tables in Sections 3 and 4, and the plot of Figure 1.1.

One of the reasons such organization of IIL methods does not exist so far is due to the varied terminology used by different authors to refer to some of these methods, which in many cases, only partially overlap. Below, we introduce most of the names and keywords used to refer to the approaches that are relevant in this work.

1.2 Terminology Unification

In the literature, there are many terms linked to ML approaches that enable teachers to interactively shape learning systems. As a consequence, many of them are used to describe similar learning problems,

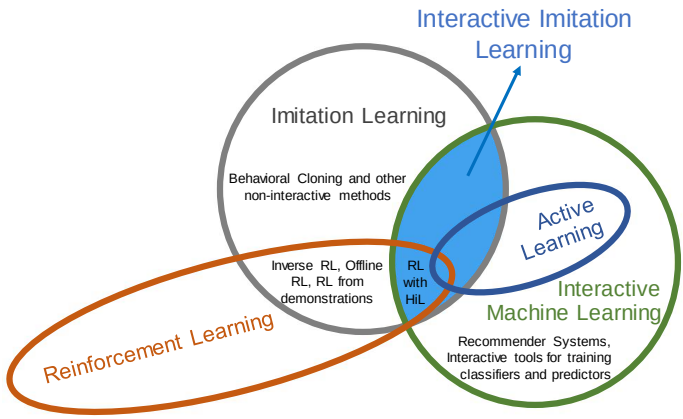


Figure 1.2: Relationship between different sets of learning paradigms related to the scope of this work. The intersection of IL with Interactive Machine Learning (IML)(blue area) is what defines the scope of this work, called here IIL

which makes it difficult for practitioners (especially beginners) to have a clear outlook of the field when studying the well-spread collection of related papers. In this section, we introduce some of those terms and discuss how they relate to each other, group them into sets that partially overlap or contain some others, and provide a definition of IIL. Based on this definition and structure, we set the bounds of the topic of interest of this work.

Figure 1.2 depicts with a Venn Diagram the relationship between all learning paradigms discussed below.

1.2.1 Imitation Learning

In the context of robotics, the terms Learning from Demonstration (LfD), Programming by Demonstrations or Programming by Doing (PbD), and IL are indistinctly used when referring to the paradigm of enabling robots to derive controllers from human demonstrations (Billard and Grollman, 2013). Originally, these terms have been used by multiple authors referring to learning approaches that derive policies from datasets of explicit teacher demonstrations of a task.

Some recent methods enable human teachers to train robots through evaluative feedback, like Learning from Critique (LfC), or Interactive Reinforcement Learning (Interactive RL), in which the teachers provide feedback that rates the desirability of the exhibited behavior during training time. Although these approaches do not fully fit the literal meaning of LfD or IL, some authors consider that evaluative feedback is just one of the demonstration modes a teacher could use within a learning process (Chernova and Thomaz, 2014), therefore they also can be considered part of the world of IL.

Since IL is used at different levels of robot control and similar problems, we can rephrase the definitions of LfD, PbD, and IL as *the set of ML methods that leverage teacher's input as the source of knowledge for training sequential decision-making systems.* Most of the time, the teacher is a human user, while in some cases it could be another decision-making agent (e.g., a computationally expensive policy like an Model Predictive Control (MPC) or a planner system), and it has an understanding of either what are the objectives of the task, what to do, how good an action/policy is, or how good is the policy with respect to others.

In other words, methods are not considered IL if they leverage the input of a teacher to train non-sequential decision-making systems, e.g. an image classifier (Fails and Olsen Jr, 2003).

In the last two decades, articles have been published reviewing varied perspectives of IL, proposing categorizations for organizing the types of methods, identifying the benefits and drawbacks of the most known approaches, listing the open challenges, and introducing and structuring the field of study (Billard *et al.*, 2008; Argall *et al.*, 2009; Billing and Hellström, 2010; Billard and Grollman, 2013; Chernova and Thomaz, 2014; Amershi *et al.*, 2014; Billard *et al.*, 2016; Hussein *et al.*, 2017; Lee, 2017; Calinon, 2018; Osa *et al.*, 2018; Li *et al.*, 2019a; Zhang *et al.*, 2019b; Ravichandar *et al.*, 2020).

1.2.2 Interactive Machine Learning

There exists a considerable amount of learning methods that leverage human teachers within the learning loop for training sequential and

non-sequential decision-making systems. Through different types of interaction, they make use of the knowledge a human has about the process, without the need to hard-coding it. Therefore, these methods enable users who are not expert ML practitioners to train models according to their insights and intuition. The set of approaches that cover all the learning loop schemes involving humans transferring knowledge to the agent is known as IML (Amershi *et al.*, 2014; Fails and Olsen Jr, 2003; Ware *et al.*, 2001; Holzinger, 2016; Dudley and Kristensson, 2018; Jiang *et al.*, 2019).

Holzinger (2016) define *"IML-approaches as algorithms that can interact with both computational agents and human agents and can optimize their learning behavior through these interactions"*. Dudley and Kristensson (2018) explain the contrast between IML and classical ML as *"Interactive Machine Learning is distinct from classical machine learning in that human intelligence is applied through iterative teaching and model refinement in a relatively tight loop of set-and-check. In other words, the user provides additional information to the system to update the model, and the change in the model is reviewed against the user's design objective"*.

Some other authors refer to the same domain with a more explicit name like Human in the Loop Machine Learning (HIL-ML) (Xin *et al.*, 2018; Wu *et al.*, 2021). Other authors refer to it in a more general way, combining the term Artificial Intelligence (AI), e.g., with Human in the Loop Artificial Intelligence (HIL-AI) (Zanzotto, 2019), or Interactive Artificial Intelligence (IAI) (Wenskovitch and North, 2020). Human Centered Machine Learning (HCML) or Human Centered Artificial Intelligence (HCAI) is a larger domain that contains all the mentioned approaches with a human in the learning loop, additionally, it also includes the approaches based on ML/AI that have humans in the execution loop, i.e., systems that interact with humans as in ML/AI-based Human-Computer Interaction (HCI) or Human-Robot Interaction (HRI) systems.

Methods of IML serve a wide domain of applications, including classification, regression, image processing, information retrieval, anomaly detection, among other systems (Ware *et al.*, 2001; Fails and Olsen Jr, 2003; Amershi *et al.*, 2012; Ngo *et al.*, 2014; Amershi *et al.*, 2014; Dudley

and Kristensson, 2018; Jiang *et al.*, 2019). It is important to clarify that although IML methods always include a human in the learning process, in some applications the human does not always perform as a *teacher*, but rather is a user about whom the system learns through the interactions without explicit signals, as it is the case for Recommender Systems (Burke, 2002; Bobadilla *et al.*, 2013; Beel *et al.*, 2016).

Active Learning is one of the most traditional approaches of IML, which consists of endowing the learner with capabilities for querying the teacher for more data in specific situations. The learner is able to choose from which data samples it learns, allowing it to learn with higher accuracy from fewer samples (Cohn *et al.*, 1996; Settles, 2009).

1.2.3 Interactive Imitation Learning

The set of IML covers a broad spectrum of problems it can be applied to, including sequential and non-sequential decision-making. IL is narrower and specific to sequential problems. Unlike IML, IL also involves methods that learn from teachers in a sequential manner, without the need for continuous interaction in the learning loop, as is the case of Behavioral Cloning (BC), Inverse Reinforcement Learning (IRL), offline RL, or RL from demonstrations, which learn from a set of demonstrations that have been recorded before the learning process starts.

Also known as direct IL, BC (Bain and Sammut, 1995) applies supervised learning to a set of previously recorded expert demonstrations, in order to obtain a model that imitates the demonstrations. In contrast, IRL is known as indirect IL because it uses recorded demonstrations to obtain an objective function or reward function that explains the goal of the task, so it can be used in an RL process for obtaining a policy that imitates the demonstrator (Ng and Russell, 2000; Zhifei and Joo, 2012). In offline RL the principles of classical online RL are extended to be applied over datasets of demonstrations, without collecting any new sample during training time (Levine *et al.*, 2020). We refer to RL from demonstrations to the domain of all methods of classical online RL that leverage recorded demonstrations to initialize the policy, or that keep that data in a buffer that is continuously used for updating the policy along with the new samples that are collected with the interactions (Kober and Peters, 2008; Hester *et al.*, 2018).

The previous methods are not interactive, even though they learn from data demonstrated by teachers. We hereby, take the term IIL that has been previously used in the literature and redefine it as the set of methods resulting from the intersection of the IL and IML sets. Therefore, we can say that *IIL methods involve the approaches that learn from the knowledge provided by a teacher in the learning loop of a sequential decision-making system.* Human teachers can transfer their knowledge to the learning agent through different modalities of interaction, and they are able to observe the effect of their feedback throughout the incremental learning process.

Methods of ML that actively choose or query training samples are known as Active Learning (Settles, 2009) methods, and they aim to increase the sampling efficiency of the learning process. It is a subset of IML that also overlaps with the IIL domain.

It is important to make a distinction between IIL, IML, and Interactive Learning Systems (ILS), which is also used in the literature and sometimes referred to as learning from interactions, or interactive learning. ILS are real/virtual entities that learn from the interaction with the world, a human, or another entity. This definition is complemented in Cuayáhuitl *et al.* (2013) with the description: *"A machine can therefore be said to learn from interactions in a particular class of tasks if its performance improves with the given interactions over time"*. The ILS that learn from the interaction with the world/environment enclose RL methods (Sutton and Barto, 2018), wherein the agent learns from its own experience and not from a teacher. The subset of ILS that learn from the interaction with other agents acting as teachers results in the same set of IIL methods, which are the focus of this work.

RL systems that obtain data from human teachers in the form of either demonstrations or evaluations (human reinforcements) during the learning process are known as Human in the Loop Reinforcement Learning (HIL-RL) and are also a type of IIL.

1.3 Others Surveys and Outline

In recent years, there has been an explosion in the adoption of IL methods. There exist a large body of surveys discussing IL from different

points of view. In particular, Chernova and Thomaz (2014) provides a general overview of the methodology of learning from demonstration where different topics are analyzed, such as how the human teacher interacts with the robot to provide demonstrations, which modeling technique to choose (low/high level), how the human can refine an existing task and how to incorporate interactive and active learning components. Given the big spectrum of the paper of Chernova and Thomaz (2014), interactive methods are mentioned as one possible evolution of IL, but they are not the main focus of the work, and, therefore, not analyzed in depth.

A similar collection and analysis of the literature were conducted recently by Ravichandar *et al.* (2020). Here, topics such as non-expert robot programming, data efficiency, safe learning, and performance guarantees are discussed. The authors highlight the importance of learning from social cues, reasoning about the availability of human demonstrators, how to behave in their absence and how to ask for help. However, IIL is only marginally analyzed.

Similarly, Hussein *et al.* (2017) propose a survey on different learning methods for IL. The survey underlines how BC has limitations due to errors in the demonstration and poor generalization. As a possible solution, it is proposed to combine IL with RL, refine the policy with RL, or use active learning. However, marginal attention is given specifically to interactive methods.

In a recent survey, Osa *et al.* (2018) provide a structural analysis on IL, focusing on BC and IRL methods. The authors mention that incremental and interactive learning methods can be employed to alleviate the *covariate shift* problem (Section 2.2.6) that exist in BC methods. While they highlight the necessity of such methods from an algorithmic and mathematical perspective on machine learning, the authors do not provide an extensive treatment of the topic, as it is outside the scope of their work.

The topic of Human-Centered RL is investigated by (Li *et al.*, 2019a) as well as Zhang *et al.* (2019b), where human evaluative feedback is used to teach behaviors to learning agents. They divide the field into three categories: learning from human reward, from interpreted human reward, and from action-translated human reward. Although these works are

surveying the concept of human feedback from a RL perspective, a broader discussion of other IIL methods is not covered.

In our work, we provide a survey of the Interactive Imitation Learning literature, ranging from seminal early work to the most recent advances. We investigate the role of IIL in the broader picture of sequential decision-making problems, with a focus on robotics applications. Besides providing an organized view of the state-of-the-art of the field, we aim to distill the most important takeaways and contribute a useful perspective on the topic. Our goal is for this manuscript to be a helpful reference for future work as well as a starting point for newcomers to the field. Our discussion spans multiple dimensions, ranging from the type of feedback a human teacher can provide to the agents, to the models that are learned through this interaction, to the existing benchmarks and applications proposed in recent years. In particular, we structure the analysis over multiple sections as follows:

- Section 2 provides an overview of the sequential decision-making problem and its different formulations, formalizes the IIL problem and defines core concepts such as Feedback and Covariate Shift.
- Section 3 discusses the different modalities of feedback that a human teacher can provide to the robot, ranging from evaluative to preference to corrective feedback or interventions. We examine their strengths and weaknesses, with a focus on the trade-off between richness of information and human effort required.
- Section 4 considers the various types of models that the robot is able to learn from the provided feedback, including policies, transition models, and objective functions. We discuss how certain models are best learned by specific types of feedback, and how they are used to achieve the main objective of solving sequential decision problems.
- Section 5 reviews auxiliary models that the robot could learn in addition to the main objective, such as uncertainty and risk estimation models, environment dynamics, task features and models of the human teacher. We analyze the advantages that such models provide and the settings in which they can be adopted.

- Section 6 discusses the different types of function approximation and model representation strategies common in the literature, including motion-conditioned models and deep neural networks. We consider their advantages and disadvantages and provide recommendations on their usage.

- Section 7 provides a comparison between on-policy and off-policy methods with a focus on the IIL setting.

- Section 8 analyzes the special case of IIL methods used in glsrl framework, called RL with Human in the Loop.

- Section 9 presents an overview of the interfaces used for enabling the communication between the robot/computer and the teacher, examining their role and importance in the learning pipeline. They range from physical contact with the robot embodiment to external devices such as remote controllers to contact-free approaches such as video and voice.

- Section 10 provides an overview of the human factors to consider in IIL, such as available human-robot interfaces, user experience, and performance metrics, as well as guidelines on how to design user studies in IIL.

- Section 11 surveys the principal benchmarks and datasets used in the literature to evaluate the proposed methods as well as the different fields of application of these algorithms, such as assistive, household, medical or industrial robots;

- Section 12 provides a discussion of the current challenges and opportunities in the field of IIL, as well as directions for future work.

- Section 13 completes the survey with a summary of the main concepts discussed as well as the most relevant takeaways and contributions to the field.

2

Theoretical Background

In this section, first we formalize the sequential decision-making problem using the Markov Decision Process (MDP) framework (Bellman, 1957), and then formalize the IIL problem.

Many problems like solving Rubik's cube with a robotic hand, controlling the propulsion of a rocket, swinging up a pendulum or finding the best strategy in a chess game share the necessary idea of finding the best set of actions that would successfully accomplish the task. These problems share many properties and therefore they can be modeled using a common framework (i.e MDP).

2.1 Decision Theory

A wide variety of problems can be formalized as a sequential decision-making process, where the decision-making authority is an *agent*, operating in a certain *environment*. At each time instance t (also known as time step), the agent receives information describing the situation of the environment with the *state* vector s_t, and executes an *action* a_t, aiming to change the environment towards a desired state according to the goal of the task. The environment transitions to a new state s_{t+1}, and provides a reward r_t, which is a signal that explains the objective of the task.

When a decision-making problem has well-defined initial and terminal conditions, it is known as a *finite horizon* problem, and the period of time between its start and end is called an *episode*. The collection of states and actions experienced by the agent throughout an episode is known as a *trajectory* $\tau = (s_0, a_0, ..., s_T, a_T)$, where T corresponds to the number of time steps visited by the agent.

Decision theory provides a formal and complete framework for decision-making by combining probability and utility theory (Russell and Norvig, 2016).

2.1.1 Markov Decision Process (MDP)

Initial foundations for MDP are set by Bellman (1957) and further extended by Howard (1960). An MDP models a stochastic, sequential decision-making process in a fully observable environment as a tuple $< \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} >$ with four components:

- $\mathcal{S}$: A set of all possible *states* s.
- $\mathcal{A}$: A set of all possible *actions* a. Some problems may have a state-dependent set of actions ($\mathcal{A}(s)$).
- $\mathcal{T}(s' \mid a, s)$: A *transition model* that defines $\mathbb{P}(s_{t+1} = s' \mid s_t = s, a_t = a)$, probability of reaching state s' if action a is applied in state s ($\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$).
- $\mathcal{R}(s, a)$: A *reward function*, determining the *reward* r received for applying action a in state s ($\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$).

Decisions are modeled as state-action pairs (s, a). The next state s' is determined by a probability distribution, which is defined by the transition function $\mathcal{T}$ and it is based on the current state s and the applied action a. The Markov property defines that the next state s' is dependent only on the current state and action, while the previous states and actions do not have any influence on the current transition. A deterministic policy π is a mapping from states to actions, defining which action should be chosen in that state in $\mathcal{S}$ ($\pi : \mathcal{S} \mapsto \mathcal{A}$). The

policy can also have a stochastic representation, with a distribution over state-action pairs ($\pi : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$).

In certain problems, the agent cannot directly observe the underlying state. Instead, the state can only be inferred indirectly, using an observation model (the probability distribution of different observations given the underlying state). For such problems, it is appropriate to consider the Partially Observable Markov Decision Process (POMDP) (Kaelbling *et al.*, 1998), where the dynamics are still described using MDPs, and the additional observation model $O(s, o)$ specifies the probability of perceiving an *observation* o in a state s.

2.1.2 Sequential Decision-Making Problem

The objective of solving decision-making problems modeled with MDPs or POMDPs is to find the policy π^*, that maximizes the expected accumulated reward (also known as *utility* or *value*), i.e.,

$$\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}_{\tau \sim p_\pi(\tau)} \left[\sum_{t=0}^{T} \gamma^t \mathcal{R}(s_t, a_t) \right], \tag{2.1}$$

where $p_\pi(\tau)$ corresponds to the trajectory distribution induced by π, Π to the set of possible policies, and γ is a discount factor. Since problems, in general, can have *infinite horizon* ($T \to \infty$), this sum could diverge. Therefore, the sum can be discounted with the discount factor γ, where $0 \leq \gamma < 1$. This problem formulation assumes the reward hypothesis, which claims that *"all of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received scalar signal (reward)"* (Sutton and Barto, 2018).

There exists a plethora of methods for solving sequential decision-making problems, each with unique assumptions, strengths and weaknesses. We make rough distinctions between three different approaches: *Control*, *Planning* and *Learning*. The boundaries of these approaches are not always clear as many practical solutions commonly lie in between or combine these approaches. In this work, we focus on Interactive Robot Learning approaches, where the goal is to devise a policy by learning from interactions with humans.

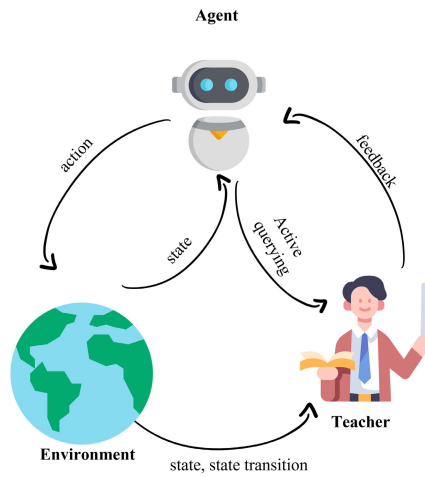


Figure 2.1: IIL learning loop.

2.2 Interactive Imitation Learning

2.2.1 MDPs in IIL

In IIL, a human teacher, which we will refer to as *the teacher*, aims to improve the behavior of a learning agent, which we will refer to as *the learner*, by occasionally providing feedback to it as a function of the observed behavior (see Figure 2.1). The period of time when the teacher provides feedback to the learner is known as the *learning process*, which finishes whenever the human considers the learner's behavior appropriate or when no more improvement is observed. The human feedback can be modeled with the *feedback function* $\mathcal{H}$. Although $\mathcal{H}$ can evolve throughout a learning process (i.e., a human may modify its understanding of a task when teaching), for simplicity, the following of this section assumes this function does not change.

$\mathcal{H}$ is presented as a more general alternative to the reward function employed in the MDP framework. At every time step, as a consequence of the agent's behavior, $\mathcal{H}$ outputs a *feedback signal* H_t, which is defined as any type of information that can be used to improve the agent's policy

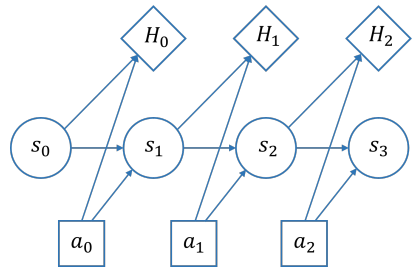


Figure 2.2: MDP with the feedback signal H_t.

(see Figure 2.2). In IIL, feedback can be occasional; therefore, H_t consists of two values: $h_t \in \mathbb{R}^n$ and $g_t \in \{0, 1\}$. h_t provides the information employed to improve the agent's performance and g_t indicates the instances where feedback was given, i.e., h_t exists whenever $g_t = 1$. Furthermore, and differently from the reward function in RL, $\mathcal{H}$ may depend on previously visited states $s_{\leq t} \equiv (s_0, s_1, ..., s_t)$ and actions $a_{\leq t} \equiv (a_0, a_1, ..., a_t)$. Hence, in the deterministic case, $\mathcal{H}(s_{\leq t}, a_{\leq t}, s'_t) : \mathcal{S}^{t+1} \times \mathcal{A}^{t+1} \times \mathcal{S} \mapsto \mathbb{R}^n \times \{0, 1\}$ (note that the domain can be a subset of $\mathcal{S}^{t+1} \times \mathcal{A}^{t+1} \times \mathcal{S}$, where X^t represents X to the power of t). Alternatively, $\mathcal{H}$ can be modeled as a probability distribution $\mathcal{H} : \mathcal{S}^{t+1} \times \mathcal{A}^{t+1} \times \mathcal{S} \times \mathbb{R}^n \times \{0, 1\} \mapsto [0, 1]$. Finally, note that this formulation can be extended to cases where the agent generates active queries, where $\mathcal{H}$ would also depend on them.

Given that h_t does not necessarily represent a reward, the problem formulation of the MDP needs to be modified accordingly. The next subsection discusses how to approach this problem.

2.2.2 Interactive Imitation Learning Objective

The goal of sequential decision-making problems is to find a policy π that generates trajectories $\tau \sim p_\pi(\tau)$ such that an objective function $J(\pi)$ is minimized. In RL, for instance, the objective function is defined by the policy's (negative) expected return. In IIL, however, there is not always direct access to this function, as it is commonly represented implicitly inside the teacher's mind and, therefore, it is not always possible to minimize it directly. Consequently, more generally, it is

possible to formulate the problem in terms of an observable *surrogate loss* $L(\pi, \mathcal{H})$ computed as a function of the feedback function $\mathcal{H}$. We assume that the minimization of $L(\pi, \mathcal{H})$ indirectly minimizes $J(\pi)$ (or at least leads to near-optimal solutions). Note that when the true objective function of the problem is available, these two functions are the same (i.e., $L = J$). Hence, IIL aims to find a *learner's policy* π^l by solving the following optimization problem:

$$\pi^{l*} = \arg\min_{\pi \in \Pi} L(\pi, \mathcal{H}). \tag{2.2}$$

One key aspect of this equation is the approach employed to search through the space of solutions Π. In practice, when learning this sequential decision-making problem, the data used to optimize Equation 2.2 comes from a policy that interacts with the environment, which biases the optimization problem. Hence, depending on this policy, different solutions will be obtained.

To make this idea evident, the problem can be formulated in terms of the expected immediate cost $C(s, a)$ of performing an action a for a state s (Ross *et al.*, 2011). Then, we can express this cost in terms of a policy π with $C_\pi(s) = \mathbb{E}_{a \sim \pi(s)}[C(s, a)]$. Consequently, the objective function becomes the accumulated expected immediate cost $J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{s \sim p_\pi^t(s)}[C_\pi(s)]$, where T corresponds to the task horizon and $p_\pi^t(s)$ is the state distribution at time step t induced by π. Once again, given that we might not have access to $C_\pi(s)$, the problem can be formulated in terms of the immediate expected surrogate loss $\ell_\pi(s) = \mathbb{E}_{a \sim \pi(s)}[\ell_\pi(s, a, \mathcal{H}(s, a))]$, yielding the following IIL optimization problem:

$$\pi^{l*} = \arg\min_{\pi \in \Pi} \sum_{t=0}^{T} \mathbb{E}_{s \sim p_\pi^t(s)}[\ell_\pi(s)]. \tag{2.3}$$

In practice, the expected value of the surrogate loss in Equation 2.3 is estimated from the data collected by a policy that interacts with the environment (i.e., p_π^t is induced by this policy). For instance, BC methods use the *teacher's policy* π^h to collect training data (i.e., $s \sim p_{\pi^h}^t(s)$), or, in other words, the data comes from executions of the task performed by the teacher.

It turns out that methods like BC that learn from data gathered with a policy different from the one that is later evaluated (i.e., π^l) suffer from *covariate shift*. In this context, covariate shift means that the distribution of states visited at evaluation time by the learner (i.e., $s \sim p^t_{\pi^l}(s)$) differs from the one from where the training data was sampled from (e.g., $s \sim p^t_{\pi^h}(s)$). As a consequence, the learner visits states that were not well represented in the training data, leading, possibly, to catastrophic mistakes. For more details regarding covariate shift, we refer the reader to Section 2.2.6.

Therefore, in IIL, the training data distribution depends on the learner's policy. In this way, these methods aim to minimize the state distribution mismatch between the data sampled at training and test time. Nevertheless, this poses a *chicken-or-the-egg* problem, since without knowing the learner's policy in advance, it is not possible to generate data from the trajectories that this policy would generate (Ross and Bagnell, 2010). IIL methods address this by solving the problem iteratively, i.e, the learner's policy is used to collect data, improve its behavior from the data and repeat this process N times until a well-performing policy is obtained. Hence, by noting that $\sum_{t=0}^{T} \mathbb{E}_{s\sim p^t_\pi(s)} [\ell_\pi(s)] = T\mathbb{E}_{s\sim p_\pi(s))} [\ell_\pi(s)]$, where $p_\pi(s) = \frac{1}{T}\sum_{t=0}^{T} p^t_\pi$ corresponds to the average distribution of states (Ross *et al.*, 2011), the general IIL problem can be formulated as:

$$\textbf{IIL problem: } \pi^{l*} = \arg\min_{\pi\in\Pi} \sum_{i=1}^{N} \mathbb{E}_{s\sim p_{\pi^l_i}(s))} [\ell_\pi(s)] \,. \tag{2.4}$$

Note that in this equation there is an abuse of notation, as $p_{\pi^l_i}(s)$ represents a distribution of states that *depends* on π^l_i, but the actions taken for collecting training data do not always necessarily have to distribute exactly as π^l_i.

From Equation 2.4 it can be observed that every IIL method has the following properties:

1. A surrogate loss ℓ_π is computed as a function of the feedback function $\mathcal{H}$.

2. The problem is formulated over state distributions that depend on the learner's policy.

3. The problem is solved iteratively by sampling, at each training iteration, from state distributions that depend on the current learner's policy.

2.2.3 Episodic Feedback

A family of IIL methods solves Equation 2.2 by solving the inverse problem, i.e., $L(\pi, \mathcal{H})$ is unknown and human feedback is employed to estimate $\hat{L}(\pi, \mathcal{H})$. Then, this estimation is minimized $\hat{L}(\pi, \mathcal{H})$ with some optimization method (e.g., path planning or RL). Commonly, several trajectories are sampled from the learner's policy π^l to get Monte Carlo estimates of $L(\pi, \mathcal{H})$ and feedback is provided at the end of them, i.e, $g_t = 0$ the rest of the time. This feedback consists of an evaluation over the complete trajectory that has the form of a choice/preference (Wilde *et al.*, 2022; Wilson *et al.*, 2012; Christiano *et al.*, 2017), i.e., at each iteration, given the execution of two or more trajectories from the learner, the teacher provides a ranking of them. Then, the feedback is employed to gradually shift the trajectories generated by the learner in a direction where their performance will increase.

2.2.4 Per Step Feedback

Alternatively, many IIL methods directly solve Equation (2.3) following approaches that were derived either from RL (value maximization) or the classical IL (divergence minimization) literature. Therefore, in these cases, feedback is provided in a *per-step basis*, i.e., the teacher observes the behavior of the learner at each time step and provides feedback if necessary.

2.2.5 Value Maximization

Value Maximization methods correspond to IIL approaches that employ human feedback to solve problems formulated using the RL approach (see Equation 2.1). In other words, some part of the RL problem is modified through $\mathcal{H}$.

The most direct way of doing this is by *naively* replacing the reward function of an existing RL approach with $\mathcal{H}$ and executing the learning process as if nothing changed. However, prior research has shown that such methods may induce *positive reward cycles*, which could lead to unintended behaviors (Ho *et al.*, 2015). This shortcoming lead to the development of approaches that built upon the RL literature but take into account this and other limitations in the method design. For more information regarding these methods, the reader is referred to Sections 3 and 8.

Divergence Minimization

The IIL methods that are derived from the literature of classical IL can be modeled as a divergence minimization problem where we assume that we have access to expert trajectories from π^h. Then, the problem is modelled as minimizing the distance between the trajectory distribution of the expert/human $p_{\pi^h}(\tau)$ and the learner $p_{\pi^h}(\tau)$. The $f-divergence$ family (Liese and Vajda, 2006) is a class of divergences that measure distances between probability distributions. Hence, the IL problem can be seen as an *f-divergence minimization problem* (Ghasemipour *et al.*, 2020; Ke *et al.*, 2020). By denoting the f-divergence between two distributions as $D_f(\cdot, \cdot)$, IL can be formalized as:

$$\pi^{l*} = \arg\min_{\pi \in \Pi} D_f \left(p_{\pi^h}(\tau), p_{\pi}(\tau) \right). \tag{2.5}$$

BC methods solve Equation 2.5 by using the *forward Kullback–Leibler divergence* (KL), which reduces the problem to the Maximum Likelihood Estimation (MLE) of the teacher's policy from samples drawn from the trajectory distribution induced by the teacher's policy (Bishop, 2006), i.e.,

$$\pi^{l*} = \arg\max_{\pi \in \Pi} \mathbb{E}_{\tau \sim p_{\pi^h}(\tau)} \left[\sum_{t=0}^{T} \ln \pi(a_t | s_t) \right]. \tag{2.6}$$

Interestingly, if the *Total Variation* (TV) distance between these distributions is minimized instead, the problem reduces to the minimization of the forward KL divergence between the teacher and the learner

policies from a state distribution that follows the learner's policy (Ke *et al.*, 2020)

$$\pi^{l*} = \arg\min_{\pi \in \Pi} \mathbb{E}_{s \sim p_\pi(s)} \left[D_{KL} \left(\pi^h(a|s), \pi(a|s) \right) \right]. \tag{2.7}$$

Notably, if we define the surrogate loss of an IIL problem as $\ell_\pi = D_{KL} \left(\pi^h(a|s), \pi(a|s) \right)$, then we would have an IIL method that minimizes the TV divergence between the teacher's and the learner's policy. The method Data Aggregation (DAgger) (Ross *et al.*, 2011) minimizes this objective function, which inspired a broad family of IIL methods.

In this case, the feedback function directly outputs a desired action for a given state, i.e., h_t corresponds to a sample from $\pi^h(a|s)$. The samples h_t can be employed to estimate π^h by solving the MLE problem. Alternatively, π^l can be modeled as a deterministic policy. In such cases, the samples are approximated by the minimization of a distance between π^l and h_t (e.g., Mean Squared Error (MSE) minimization). This approach can indirectly solve Equation (2.7) if some assumptions are made; for instance, if it is assumed that π^l follows a Gaussian distribution with fixed variance, solving Equation (2.7) is equivalent to finding a discrete policy that models the mean of this distribution through MSE minimization (Osa *et al.*, 2018).

2.2.6 Covariate Shift

One of the main advantages of using IIL over offline IL methods is its data efficiency. Here, data efficiency is evaluated as the amount of human data (i.e., feedback) that is required to obtain well-performing policies (according to the human's judgment). Offline IL methods require large amounts of data because of covariate shift. Covariate shift is defined as the prediction problem where the source and target domain probability densities are different (Sugiyama, 2015; Osa *et al.*, 2018) and, therefore, the learned model is required to make predictions with inputs that diverge from the distribution of the training data.

In the context of offline IL, the source domain corresponds to $p_{\pi^h}(\tau)$ and the target domain corresponds to $p_{\pi^l}(\tau)$, i.e, the learner learns from samples that are drawn from the teacher's policy. However, given that

the learning problem will always be subject to errors, at prediction time, the states that the learner visit will gradually diverge away from the states presented in the training data, leading the agent to visit unknown regions and, therefore, to possibly make catastrophic mistakes. To solve this problem, it is necessary to collect enough data such that the state space is covered as much as possible, which in complex problems can become intractable or require large amounts of resources.

Alternatively, in IIL, the learner collects data following its own distribution $p_{\pi^l}(\tau)$. Hence, by learning iteratively following this strategy, the agent learns to correct its mistakes and avoid regions that may be dangerous or that will not lead to completing the task. In this case, it is not necessary to collect data over the complete state space, but only in the regions that π^l visits, and, therefore, fewer data is required to obtain well-performing policies.

In the next section, different methods that solve either the Value Maximization or the Divergence Minimization problem, along with the type of feedback (i.e., *feedback modality*) that they employ, are introduced.

3

Modalities of Interaction

In the IIL literature, there exist various modalities of interaction that a human teacher can adopt to communicate with the learning agent. In this section, we aim to provide a classification of these methods by answering the question *what kinds of feedback could a teacher use to train an agent interactively?* The feedback is the signal containing the information that human teachers explicitly communicate to the learning agent through a Human-Robot (or Human-Computer) interface. Different kinds of feedback are useful for transferring knowledge to the agent depending on factors like the task complexity, the teacher's understanding or expertise about it, the potential of the teacher to learn through the training process, or the available interface for providing feedback.

The short answer to that question provides two main categories that group the learning methods. They are based on the domain of the feedback provided by the teacher, which could be either in the *evaluative space* or in the *transition (state-action) space*. The former covers the methods in which the teacher provides a signal of assessment or evaluation about *how well* the agent performs, while the latter category gathers the methods that require the teacher to provide feedback that let the agent learn *how to do* the task.

Table 3.1: Modalities of interaction according to the kind of feedback.

Learning From Human	**Absolute Feedback**	**Relative Feedback**
Feedback in Evaluative Space	Reinforcements	Preferences
Feedback in Transition (State-Action) Space	Absolute Corrections	Relative Corrections

In both categories, there are two ways for the user to transmit the assessment or guidance to the agent. The teacher could provide feedback that is either relative or absolute. In the relative feedback case, the teacher provides a signal that contains information about the direction the agent behavior should shift to, with respect to current or other policy executions, e.g., how good a policy/transition is with respect to others, or how a transition should be modified with respect to the current one. However, since it is only a relative direction, it does not specify explicitly what the exact input-output mapping is that the model should fit to, and it might be required to gather many feedback samples to tune the final mapping, even for a specific state or state-action pair. On the other hand, the absolute feedback contains information about the current execution regarding the optimal behavior, implicitly known by the teacher. The relative feedback requires a lower cognitive load (i.e., less mental effort) for teachers because it is less informative than the absolute counterpart, which makes it in some cases less data efficient. In other words, the use of relative and absolute feedback can represent a trade-off between data efficiency and cognitive load of the teachers during the interaction.

Table 3.1 presents the four modalities a teacher can use for interacting with a learning agent, depending on the kind of information provided and the way it is represented (absolute or relative). Methods corresponding to each row will be discussed below in Sections 3.1 and 3.2 respectively.

3.1 Human Feedback in Evaluative Space

We first review different approaches that provide the agent with evaluative feedback, which consists of a scalar value indicating the quality of the agent's behavior. This family of methods tends to be confused with the set of RL with Human-in-the-Loop (RL-HiL) methods since they partially overlap, however, the latter group is the result of different classification criteria. In Section 8, we focus to review the RL-HiL methods, which we define as the interactive methods that combine RL and human input, and learn from both the reward function of the environment and the human feedback that can be of any kind of the modalities discussed in this section, i.e., they are not restricted to learn only from human evaluations.

The earliest works in interactive learning belong to the evaluative feedback category and were inspired by animal clicker training, a common strategy used to train dogs and other domestic pets (Pryor, 1999). Animal training for purposes like assistance or detection dogs is in fact proof that humans can transfer knowledge to other agents through simple signals specifying whether a behavior is acceptable or not, without explicit demonstrations of *how to do* the task, unlike in traditional methods of LfD. Depending on the method, these feedback signals could be provided to evaluate a transition in a specific time step or a complete roll-out. They could be absolute evaluations of performance, or relative as in the cases where the teacher describes how some executions are better or worse than others, using either pair-wise comparisons or rankings.

Some of the early approaches that explored the use of evaluative feedback from human supervisors to interactively train agents are Interactive Evolutionary Computation (IEC) (Takagi, 1998; Takagi, 2001; Smith, 1991), which run a Genetic Algorithm (GA) in which the fitness function is given by the human after observing the performance of the individuals of each generation, i.e., the teacher evaluates each roll-out, and those evaluations are used along with the genetic operators in the search of population (solution) improvement. Already in the '90s and the beginning of this century, these methods were applied in a very wide spectrum of applications including graphic art and animation, music,

database retrieval, industrial design, face image generation, control, and robotics, among many others (Takagi, 2001).

The use of IEC to problems in robotics (Lewis *et al.*, 1992; CWI and Amsterdam, 1997; Kamohara *et al.*, 1997; Lund *et al.*, 1998; Nojima *et al.*, 2003), known as Interactive Evolutionary Robotics (IER), also attracted the interest of the researchers in those years, however, that domain of study has not been very active in the last decade. Nevertheless, the community has been inclined to develop methods inspired by RL, with the difference of receiving interactive reinforcements from the human in the learning loop, instead of receiving them from a reward function.

Using evaluative feedback provided by a human teacher simplifies two problems compared to an autonomous learning process like RL, or an Evolutionary Strategy. It helps to bypass the difficult problem of designing an objective function used for providing feedback in the autonomous case, and additionally, the implementation of the system is simpler since the infrastructure for computing the reward is not required. However, the human evaluations can be inconsistent due to multiple external factors, or even not compliant with the MDP framework, therefore, the algorithms should take into account these considerations in order to avoid convergence issues.

3.1.1 Learning from Human Reinforcements

The methods based on absolute feedback in the evaluative space are approaches that take the human signal as a punishment or reward of the current policy execution with respect to the optimal policy implicitly known by the teacher, rather than an evaluation resulting from the comparison with other actions or policy executions (see Figure 3.1). With respect to evolutionary strategies, using human reinforcements can handle the credit assignment problem better, which is "the problem of assigning *credit* or *blame* for overall outcomes to each of the internal decisions made by a learning machine and which contributed to those outcomes" (Haykin, 2001). The feedback is closely linked to the decisions that lead to positive or negative rewards, unlike using a scalar measure that represents the fitness of an entire roll-out of a GA.

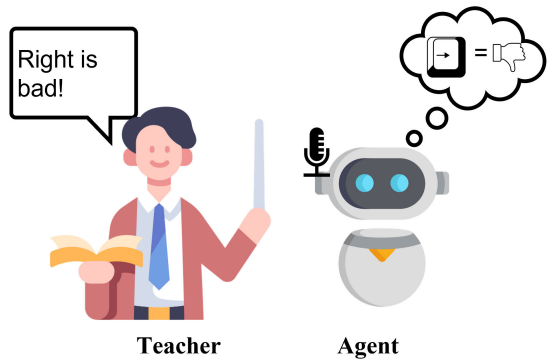


Figure 3.1: Learning from human reinforcements loop: the teacher is teaching the robot to go to the left and he gives bad rewards when it goes right.

Reinforcement Learning Only with Human Rewards

A simple way of introducing human feedback in the learning process of the agent is to adopt standard RL algorithms and let a human teacher provide the reward signal. Under this paradigm, non-expert users can teach the decision-making systems, even online, by delivering their feedback interactively as an evaluative (approval or disapproval) signal in a RL framework where the reward is completely given by the human (Kaplan *et al.*, 2002; Blumberg *et al.*, 2002; Mitsunaga *et al.*, 2008; Tenorio-Gonzalez *et al.*, 2010; León *et al.*, 2011; Suay and Chernova, 2011; Pilarski *et al.*, 2011; Yanik *et al.*, 2014; Hurtado *et al.*, 2021; Londoño *et al.*, 2022).

Thomaz and Breazeal (2006) show how Interactive RL enables a human user to provide positive and negative rewards in real-time in response to robot actions and to advise anticipatory guidance input that constrains action selection choice and guides the learner towards the desired behavior. Since a human reward may have a different meaning with respect to an encoded environment reward function, which is the basic reinforcement used in the conventional RL approaches, a series of works have analyzed how to model the human reinforcement (Thomaz, Breazeal, *et al.*, 2006; Thomaz and Breazeal, 2007a).

The TAMER Framework

An important consideration to take into account for agents learning from human rewards, in contrast to learning from environment rewards is that "MDP reward is informationally poor yet flawless and human reinforcement is rich yet flawed" (Knox and Stone, 2010). In other words, interpreting the human feedback in the same way as an MDP reward signal does not take advantage of the prior knowledge of the teacher regarding the long-term consequences of actions. The *shaping* approach allows interactively training of an agent through signals of positive and negative reinforcement, which take into account the long-term consequences and optimality of the robot's action (Knox and Stone, 2009). One of the seminal works based on *shaping* is the Training an Agent Manually via Evaluative Reinforcement (TAMER) framework (Knox and Stone, 2008; Knox *et al.*, 2012), which addresses how to use delayed human rewards in RL problems with discrete action spaces. Other works combine human rewards and MDP reward functions by applying transfer learning strategies, where both rewards were combined first sequentially (Knox and Stone, 2010), and in a simultaneous scheme (Knox and Stone, 2012), in what they refer to as TAMER+RL.

The authors of TAMER studied the use of the discount factor used with the rewards in RL (Knox and Stone, 2012; Knox *et al.*, 2012; Knox and Stone, 2013). They firstly concluded that high discount (low discount factor) performs better for human reward functions used as MDP reward. However, they presented a successful case of learning with a low discount from human reward (Knox and Stone, 2015). Knox *et al.* (2013) present the first implementation of the TAMER algorithm on a real robot.

Later on, Actor-Critic TAMER (ACTAMER) proposes an Actor-Critic approach that addresses RL problems with continuous action spaces and uses the same kind of feedback (Vien and Ertel, 2012; Vien *et al.*, 2013).

Finally, Deep TAMER (D-TAMER) (Warnell *et al.*, 2018) leverages the representational capabilities of deep learning in order to deal with high-dimensional inputs such as images. The authors show that the method is able to solve an Atari game environment in just 15 minutes of human-provided feedback.

Policy Shaping

An additional line of work is the Policy Shaping framework, initially introduced by Griffith *et al.* (2013). In this work, evaluative human feedback is used to directly update the policy, instead of being considered as a reward or value. The stochastic human policy is trained by increasing or decreasing the probability of an action in a certain state, depending on the feedback provided. Building upon the Policy Shaping framework, Loftin *et al.* (2014) and Loftin *et al.* (2016) propose a method to take into account the user feedback strategy, in particular taking into consideration different interpretations of lack of feedback from the teacher. They derive two Bayesian policy learning algorithms called Strategy-Aware Bayesian Learning (SABL) and Inferring Strategy-Aware Bayesian Learning (I-SABL), which are able to infer the trainer's strategy directly from the received feedback. This method is further extended by MacGlashan *et al.* (2014), who demonstrate how grounding of natural language commands can be learned from a human trainer providing online reward and punishment. The combination of natural language commands and human feedback for training makes the training procedure simple and intuitive for non-experts users.

Convergent Actor-Critic by Humans

Deriving from the ideas of Policy Shaping, Convergent Actor-Critic by Humans (COACHe)[1] is presented (MacGlashan *et al.*, 2017). Contrary to prior work, this method assumes that the human feedback does not follow a static rule, but tends to evolve over time, depending on the current policy of the agent. In order to take into account the dependency of the provided feedback from the policy, they consider the human feedback to be a label on the advantage function. This

[1]The original acronym of this method is COACH, however, we here call it COACHe (wherein the "e" refers to the use of evaluative feedback). This is in order to avoid ambiguities with COrrective Advice Communicated by Humans (COACHc) (that in this work we add the "c" referring to the use of corrective feedback in the action space), published earlier by Celemin and Ruiz-del-Solar (2015), and mentioned in Section 3.2.2

idea is based on the insight that the TD-error used by actor-critic algorithms is an unbiased estimate of the advantage function, which is a policy-dependent value roughly corresponding to how much better or worse an action is compared to the current policy. Moreover, instead of using the feedback to learn an approximated advantage function, it is directly applied to the policy gradient update rule. An extension of the aforementioned method is presented in Arumugam *et al.* (2019), where the approach is enhanced with deep neural networks in order to cope with high-dimensional observations such as images.

Conclusion

The use of human reinforcements allows teachers to transfer to the agent the insights of what is right or wrong in the respective time step. It requires a good level of understanding of the task but not necessarily being an expert or knowing what are the exact actions that should be taken in any state.

Since this feedback does not explicitly convey what other action should be executed in case the executed one receives a punishment, one mistaken punishment from the teacher requires many new instances of feedback to revert the wrong feedback effect, which means that these approaches are not very robust to imperfect teachers.

3.1.2 Learning from Human Preference

Learning from absolute evaluative feedback has shown great success, but it requires the human teacher to provide evaluations with respect to an absolute scale. On the other hand, methods for learning from human preference consist of comparing two or more sequences of actions and providing a preference score to the agent (see Figure 3.2), and they do not require the teacher to identify and evaluate what is the credit of the decision at each time step with respect to the success or failure of the task execution, i.e., potentially reducing teacher workload. In other words, they use relative evaluative feedback that implicitly indicates the direction in which the solution in the policy space should be shifted, such that it matches the preferences of the teacher. However, since this

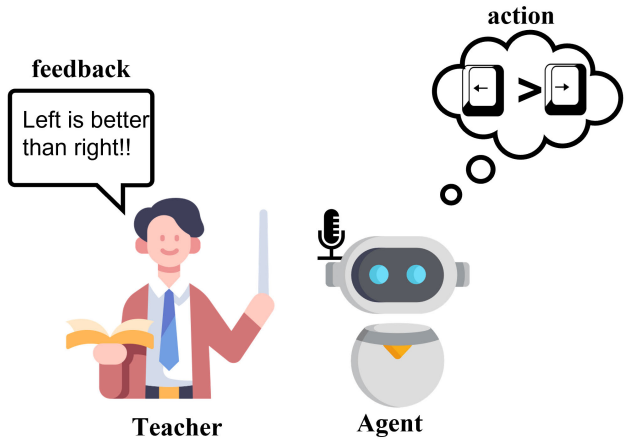


Figure 3.2: Learning from human preferences: the teacher is teaching the robot to take the left turn and it is specifically saying that going left is better than going right.

feedback is relative to other trajectories, policies, or roll-outs, it does not describe how good an execution is in general, and a policy that is preferred over another or ranked as the best out of a set of policies might be ranked low later on with respect to some different executions.

Preference-Based Policy Learning

Preference-Based Policy Learning (PPL) (Akrour *et al.*, 2011) is one of the first methods that investigate the integration between preference learning and reinforcement learning. In this work, a set of policies is shown to the human teacher who provides feedback as pairwise preferences between policies. With this information, the value of parametrized policies is estimated (here called Policy Return Estimate), the agent selects another set of policies, and the process is repeated. Similarly, Fürnkranz *et al.* (2012) consider a preference-based reinforcement learning strategy, in which a policy is leaned through an approximate policy iteration setting, instead of learning a value function of the parametrized policies. Moreover, while the former work focuses on learning a ranking function of different policies at the trajectory level, the latter considers

the preference feedback on actions in a given state. These ideas are extended by Akrour *et al.* (2014), where the agent estimates a utility function in order to account for unavoidable human mistakes. The method is evaluated on various testbeds, including a physical Nao robot, which was able to learn simple tasks such as raising a hand.

A line of work that focuses on manipulation tasks is the one presented by Jain *et al.* (2013) and Jain *et al.* (2015). The authors present a co-active online learning framework where the human teacher iteratively provides small adjustments to the trajectory currently proposed by the system. Although these methods learn from the teachers applying preference learning, the interaction modality is also with relative corrections in the transition space, as explained in Section 3.2.2, however, after recording the modified trajectory, it is compared and assumed to be preferred with respect to the one intended to be executed by the agent, hence, the data is used following the mechanisms of preference learning.

The methods presented so far show how to train robot policies by preference-based feedback, but they are limited to low dimensional states as inputs. For approaching this issue, deep Neural Networks (NNs) are incorporated into the methods, for instance, Ibarz *et al.* (2018) proposes a method where the reward function is modeled as a deep NN and is trained with a combination of demonstrations and human preferences. The key idea is that the reward function used in the RL step is not handcrafted, but rather learned via preference feedback. The combination of demonstrations and preference feedback is evaluated on multiple Atari games and shown to outperform both demonstrations or preferences used in isolation. Christiano *et al.* (2017) model both the policy and the reward function as a deep NN to deal with high dimensional observations such as images. The reward function is updated from human preference feedback, whereas the policy is updated by a traditional RL algorithm. A drawback of this approach is that the trajectories need to be sampled and replayed on screen for the user to provide feedback. Additionally, millions of steps are required for the policy to converge. Lastly, Brown *et al.* (2020) propose Bayesian REX, a fast Bayesian Reward inference algorithm from preferences. Here, ranked trajectories are used to train low-dimensional feature embeddings via a self-supervised loss. The reward function is then constructed as a linear combination of such

features. This group of approaches has been tested mainly in simulated environments, although their benefits could be extended to real-world robotic problems.

Information Maximization via Active Queries

An important component of preference-based learning methods is the choice of trajectories to compare. Some methods sample trajectories randomly from a dataset, other consider two consecutive trajectories generated by the current policy. The goal of active preference-based methods is to improve the convergence of these algorithms by generating at each step the most informative query, as measured by information theoretic metrics such as expected volume removed (Palan *et al.*, 2019). One of the early approaches in this direction is proposed by Akrour *et al.* (2012), who combine preference-based policy learning with an active ranking mechanism. Another approach for policy learning from trajectory preferences is proposed by Wilson *et al.* (2012). Here, a Bayesian model is employed in order to actively query the human teacher, and two different query selection mechanisms are investigated. The first one is called Query by Disagreement, where the main idea is to generate a sequence of unlabeled samples and evaluate them with two different classifiers. If the two models disagree on the class, then the expert is queried for a label. The second one is called Expected Belief Change, which aims to generate a set of candidate preference queries and heuristically select the best among those.

A different approach is to use the provided preferences to learn a reward function, which can then be employed for training on the downstream task, for example in a standard reinforcement learning setting. One example of this approach is presented by Daniel *et al.* (2015), who propose a framework to actively learn a reward function in a bayesian optimization setting. Sadigh *et al.* (2017) present an active learning approach, where the agent decides on the trajectories to compare by maximizing the expected information gained from the query. This information gain is modeled as the volume removed from the hypothesis space by each query. The optimization problem is solved via an adaptive Metropolis algorithm (Haario *et al.*, 2001). A novel aspect

of this work is the complex nature of the queries since it deals with continuous trajectories of a dynamical system. The authors show that this method yields faster convergence to the desired reward compared to non-active approaches. An extension of this framework is batch active preference-based learning (Biyik and Sadigh, 2018), which aims to balance the number of queries to the human teacher and the number of total interactions. This is achieved by batching multiple queries together in one request to the user. The advantage is faster iterations, and the procedure can be also parallelized when working with multiple users.

Palan *et al.* (2019) introduce the Learning Reward Functions by Integrating Human Demonstrations and Preferences (DemPref) framework, where demonstrations and preference feedback are combined to learn the weights of a linear reward function. The demonstrations are used to learn an initial prior over the space of reward functions as well as to ground the query generation process. The method is tested on different robotic manipulation tasks on a physical robot, and the additional use of initial demonstrations is shown to improve the sample efficiency of prior work. A further improvement on the active query process is provided by Biyik *et al.* (2020), which explores an information gain formulation where the ability of the human teacher to respond to a certain query is included in the optimization process. For example, if two trajectories are very similar, it might be difficult for the teacher to provide preference feedback. This approach considers the trade-off between the robot and the human uncertainty and avoids questions that would become redundant. This idea is later extended by Bıyık *et al.* (2020), where Gaussian Processes are used to model the reward function, as well as by Myers *et al.* (2022), where multimodal reward functions are learned.

Conclusion

Learning from preferences demands very low prior knowledge from the teachers since the feedback is a general performance comparison of different roll-outs, e.g., even one bit of information is enough to state the preference out of two policies, which reduces the effort of the teacher, and widens the spectrum of people who could teach a robot.

Nevertheless, this feature comes with the credit assignment problem that evolutionary-based methods have, as it was mentioned at the beginning of the section. Preferences are a relative measure of performance that evaluate a sequence of transitions, therefore the feedback does not specify what decisions make one roll-out better than the other, and the algorithm has to identify them while compromising data efficiency.

Learning from preferences methods are also sensitive to mistakes in the teachers' assessments. In both Learning from human reinforcements and preferences, the mistakes in the feedback have a negative impact on the convergence of the process, reaching lower policy performances in a longer time.

3.2 Human Feedback in Transition (State-Action) Space

Human feedback in the transition space contains information about *how to do* the task, i.e., explicit feedback that explains how a transition should be done, being it in the space of the actions, or the states. Unlike in learning from evaluative feedback, with feedback in the transition space, there is no explicit quality assessment of the policy, the feedback signals represent the teacher's insights or understanding of the task execution. This kind of feedback can be absolute, in which case the teacher is expected to demonstrate the optimal transition for the state the agent is currently visiting. Relative feedback, on the other hand, is used in cases where the teacher corrects the policy execution towards the considered right direction with respect to what the robot is executing in that time step. However, it does not assume that the correction is the optimal action, but rather a hint in that direction. The correct action is reached after some iterations that accumulate the incremental progress of many relative corrections.

3.2.1 Learning from Human Absolute Corrections

In this kind of interaction, the agents are expected to receive explicit demonstrations of the task execution by the teacher, while the learning policy is controlling the agent, as shown in Figure 3.3. Depending on the method, the teacher can provide corrective demonstrations every time

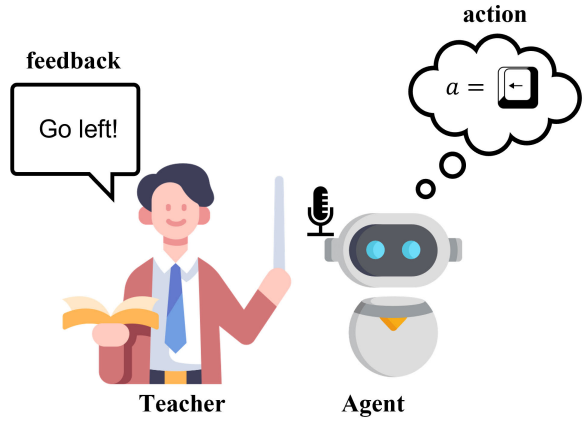


Figure 3.3: Learning from human absolute corrections: the teacher is explicitly telling the robot to go left.

step, occasionally according to the teacher's own decision, or because the learner queries them. Moreover, those demonstrations could be either only recorded, or recorded and executed. In the former case the agent executes the action from the current policy, whereas, in the latter, the agent replaces those actions with the ones demonstrated by the teacher, as in teleoperation mode. These methods are the closest to standard LfD methods like Behavioral Cloning, and some of them could even be considered a generalization of BC.

Corrective Demonstrations

One of the first approaches in this category is the Confidence-Based Autonomy framework presented by Chernova and Veloso (2009), which has two components, Confident Execution and Corrective Demonstrations. The first is a strategy that uses various thresholds to evaluate the confidence of the agent in a certain state, and in case it is too low, it stops the autonomous execution and queries the human teacher for additional demonstrations. Corrective Demonstrations are the second mechanism, which allows the teacher to provide corrections to any mistake of the agent.

The idea of Corrective Demonstrations is further investigated by Meriçli *et al.* (2010), Meriçli *et al.* (2011), and Mericli (2011), wherein they propose to leverage both prior hand-coded policies and corrective demonstrations. Instead of only obtaining directly a policy from the demonstrations, it keeps the hand-coded policy as the primary behavior, which is only replaced by the demonstrations when the robot visits states that are similar to the ones in which the corrective demonstrations were recorded. This approach was used to train the humanoid robot Aldebaran Nao to solve a complex ball dribbling task, and it shows improvement compared to a hand-coded controller.

Some of the most important methods for learning from corrective demonstrations are inspired or belong to a family of approaches based on DAgger (Ross *et al.*, 2011), which interactively records the correct action demonstrations while a novice policy is controlling the agent. DAgger was not specifically intended for human users, the teacher could be another expert policy, like a model-based controller or a planner system. Indeed, many methods have been proposed after DAgger, since the requirement of human teacher input every time step is not the most user-friendly approach.

The idea behind DAgger is to iteratively generate roll-out trajectories with the current policy, query the expert for corrections on the visited state-action pairs, and finally add the corrected actions to the dataset used for training the policy.

As with other methods in this section, this approach enables the expert to provide corrections on the states visited by the current policy, meaning that the data distribution is induced by the agent itself, drastically reducing compounding errors and distribution shift issues common in standard learning from demonstrations settings (Ross *et al.*, 2011).

DAgger requires a corrective demonstration from the teacher in every state, however, it uses a gating function based on a β probability in order to control what action is actually executed, whether the action of the learner $\pi_R(s)$ or the one of the teacher $\pi_T(s)$.

At the beginning of the learning process, β is set high for the robot to execute most of the expert teacher actions because the initial robot policies could make many mistakes that lead to dangerous or irrelevant

states. Through the iterations of the algorithm, β is decreased to zero in order to give full control to the learning agent. If $\beta = 1$ all the time, DAgger performs exactly as behavioral cloning because the data distribution is completely induced by the teacher.

If the expert is a human, this is often unfeasible and prone to incorrect labels for robotic tasks, which usually operate at high control frequency, generating a large number of actions for each trajectory. Most of the variants of DAgger (mentioned later) (Zhang and Cho, 2016; Menda *et al.*, 2019; Kelly *et al.*, 2019; Hoque *et al.*, 2021; Hoque *et al.*, 2022) differ from the original in i) the implementation of the gating function; ii) the way data is recorded, all aiming to improve workload, query efficiency, or safety.

The Svm-based reduction in Human InterVention (SHIV) algorithm (Laskey *et al.*, 2016) is very similar to DAgger, however, it actively requests labels in states considered risky, instead of requiring labels every time step, reducing the human burden. The risk is defined when previously unseen states are visited, or when the policy model has a high surrogate loss in the area of the visited state. The method was validated in grasping tasks, outperforming the original DAgger.

A possible alternative is to monitor the policy execution and intervene when necessary, taking over control from the agent completely. This is a more natural and intuitive approach for a human teacher compared to labeling individual state-action pairs. This setting can be defined as learning from human intervention, and numerous studies have been presented to investigate such methods.

There exist two main types of human intervention approaches: Human-Gated and Robot-Gated (Laskey *et al.*, 2017a). Both types change the stochastic gating function based on the probability β for executing either the action of the learner or the action of the expert, with a different strategy.

Human-Gated Interventions

Human-gated interventions allow the expert users to decide themselves when to intervene (control the agent). Its advantage is that safety is ensured by the expert, who is always ready to intervene in case of dangerous behavior.

Human Gated DAgger (HG-DAgger) (Kelly *et al.*, 2019) is a direct extension of the DAgger algorithm, where the human teacher is in charge of intervening when the agent drifts away from the desired behavior. Every time an intervention occurs, the expert trajectory is recorded and stored in the training data set used to optimize the policy. Additionally, HG-DAgger learns a safety threshold of a risk metric, which could be used as a policy confidence metric for different regions of the state space. The method is evaluated on both a simulated and a real-world autonomous driving task, showing better performance compared to behavior cloning and standard DAgger.

The assumption of the method is that the teacher does not intervene in the portions of the trajectory that are well executed. A different approach is used in the Intervention Weighted Regression (IWR) framework (Mandlekar *et al.*, 2020), where the robot's own experience is stored together with the teacher's interventions in the replay buffer. The authors show that storing such data has the advantage of reinforcing the already good behavior and improving the robustness of the policy, because more data is stored overall, and the data itself is distributed covering wider areas of the state space. Nevertheless, since the amount of intervention and non-intervention data is usually imbalanced, the authors propose a weighting parameter to prioritize the intervention samples. The method is evaluated on two challenging simulated manipulation tasks with low-dimensional observations, demonstrating better performance compared to HG-DAgger and to behavior cloning with complete demonstration.

IWR works under the assumption that the teacher is always able to correct bad behaviors, which might not be true in general, since non-expert users might be in charge of training the robot. In Chisari *et al.* (2022) the Corrective and Evaluative Interactive Learning (CEILing) framework combines human interventions with evaluative feedback. The use of evaluative feedback on non-corrected portions of the trajectory gives the human teacher the option to decide which part of the trajectory to use for training and which to discard. The method is shown to be able to train manipulation tasks from high-dimensional image observations directly in the real world in less than one hour of training.

Another related method is the Expert Intervention Learning (EIL) framework (Spencer *et al.*, 2020). EIL aims to learn from the interventions as well as from the timing of the interventions since non-intervention constitutes useful information as well. They formalize a constraint on the learner's value function, which is used to differentiate *good enough*, *bad* and *intervention* state-action pairs. The method is evaluated on a physical miniature car with a discrete action space, consisting of a library of 64 driving primitives. EIL is benchmarked against behavior Cloning and HG-DAgger, showing safe and more desirable trajectories. Another recent work in the same category is Super-Human InsErtion using Learning from Demonstration (SHIELD) (Luo *et al.*, 2021), which focuses on the problem of industrial insertion. It extends the Deep Deterministic Policy Gradient from Demonstration (DDPGfD) (Vecerik *et al.*, 2017) algorithm with a collection of different design choices, including on-policy corrections, i.e., the human can intervene to guide the agent back into the optimal region in case of deviations.

In Cycle-of-Learning (Goecks *et al.*, 2019), human-gated interventions are used for improving a policy obtained from demonstrations pre-recorded in the first stage. The experiments with a perching task using a simulated drone showed that this approach has better performance than using either only demonstrations or only human interventions.

Corrective demonstrations are not only used for learning an explicit policy, but also for learning objective functions. In Learning to Navigate from Disengagements (LaND) (Kahn *et al.*, 2021), the teacher takes over the control of autonomous navigation robots during failure situations. However, the data gathered during the interventions is not used for updating the policy, but for training a disengagement predictive model that is used as part of the cost function of the task, which is optimized during the decision-making with a model predictive control-based planner.

Robot-Gated Interventions

Robot-gated interventions require the agent to estimate when an intervention is necessary, which does not require constant attention from the teacher, since the robot is the one deciding when the intervention

should be performed, allowing the human to supervise multiple robots at once (Hoque *et al.*, 2022). These methods generally require the agent to estimate a measure of performance, safety, or uncertainty about the currently observed state, which is then used to determine when to query or enable the human teacher control. However, these kinds of approaches have to deal with the disengagement of the users, who do not react immediately when requested and require some time to be able to take over the system again.

One example of this approach is presented in DelPreto *et al.* (2020), where the policy outputs a discrete vector of confidence scores for four different gripper orientations, and the one with the highest confidence is picked. An apprenticeship model is developed, which queries the teacher intervention in case of too many failures in a row or if the output confidence is lower than a certain threshold.

A variation of DAgger called Safe DAgger (SafeDAgger) (Zhang and Cho, 2016) trains a classifier that predicts whether the learning policy deviates from the expert and, if it is the case, it switches the control to the expert in order to prevent executing unsafe actions. The authors mention that the metric used for comparing the expert and learning policy should depend on the task. Experiments with a driving simulator showed that SafeDAgger is safer and more efficient than DAgger. Ensemble Dagger (EnsembleDAgger) (Menda *et al.*, 2019)—a method that extends SafeDAgger—uses the deviation classifier as a discrepancy rule, along with a doubt rule that also switches control from the learning policy to the expert teacher. The doubt rule is computed based on the novelty/uncertainty of the policy, which is measured with the variance of an ensemble of neural networks. The doubt rule lets the agent prevent executing dangerous actions in unseen states, in addition to the actions of the learning policy that tend to deviate from the expert teacher.

The Lazy DAgger (LazyDAgger) (Hoque *et al.*, 2021) framework also extends SafeDAgger, in particular, it aims to reduce context switching by adding noise to the actions provided by the supervisor to improve the data distribution as well as by adopting an asymmetric switching criteria, modeled as a hysteresis function. Finally, Thrifty DAgger (ThriftyDAgger) (Hoque *et al.*, 2022) is proposed, where the switching

policy is learned instead. Interventions are queried in case the encountered state is sufficiently novel or risky. Similar to EnsembleDAgger, novelty is estimated by computing the variance of each output of a set of policies, whereas the "risk" of a state is estimated by learning a Q-function to evaluate the discounted probability of success from that given state and the action proposed by the policy.

Conclusion

Learning from absolute corrective demonstrations is the interactive approach most similar to standard learning from demonstration since the teacher should explicitly show what the robot has to do, i.e., she/he is required to be an expert at solving the task. However, these interactive methods have the advantage of i) reducing the compound errors, because the demonstrations are given to correct the current learning policy deviations; ii) reducing the cognitive load of the teachers since they are not required to give full demonstrations in most of the methods, but rather occasional corrections; and iii) dealing better with the mistaken demonstrations, which are not normally considered by imitation learning methods intended for non-human teachers.

Mistaken demonstrations have a direct effect that the teacher is able to predict, allowing the teachers to be aware of how to fix their mistakes. Although in most methods the mistaken feedback remains in the database used for training the policy, it is possible to compensate for them with correct labels outnumbering the mistakes, something relatively simple to do given the explicit nature of this kind of feedback (unlike with evaluative feedback).

3.2.2 Learning from Human Relative Corrections

Methods in this category do not require the teacher to know what the exact action or state transition should be applied by the agent in every state. However, they need to understand how a change of the action/state-transition magnitude would impact the execution of the task. In other words, the teacher should be able to roughly estimate how a transition would change if the policy is slightly modified. For instance,

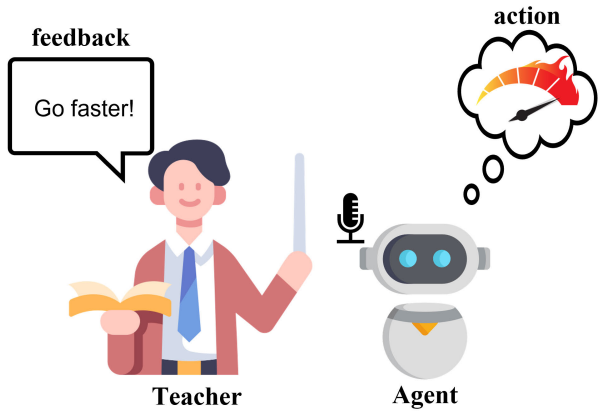


Figure 3.4: Learning from human relative corrections: the teacher is correcting the velocity of the robot telling it that it can increase the value with respect to the current one.

knowing that less power in a propeller decreases the acceleration of an aircraft or boat, or more force in the pedal brake slows down a car. With these insights, teachers could suggest how to modify a continuous policy in a more natural way (see Figure 3.4), as it happens when a coach is instructing a student for learning a physical skill, e.g., in football training: *kick the ball a bit harder and more to the right side*, in a singing lesson: *slightly increase the volume of your voice in this part of the song*, during a dance move *bend the knees less*. This correction could be discrete (increase/decrease the action) as well as continuous-valued, depending on the interface used (see Section 9)

Advice Operators One of the first interactive methods using relative corrections is Advice-Operator Policy Improvement (A-OPI), where at each iteration, it rolls out the current policy while recording the state-action pair's trajectory. Then, in an offline phase, the teacher selects the parts of the trajectory considered to be modified, along with an associated advice operator that changes the model's action of each selected pair. Finally, there is a phase of policy re-derivation based on the updated dataset (Argall *et al.*, 2008; Argall, 2009; Argall *et al.*,

2011a). An advice operator can be a relative change of the current action; for instance, in a navigation task, the advice *accelerate* would change the model's current velocity request, multiplying it by 1.1. It is a relative correction because it means *increase* the current action magnitude. An advice operator can also be the demonstration of an action, being it an absolute corrective demonstration (Section 3.2.1). For instance, the advice *stop* changes the model's velocity request to zero. Corrective demonstrations and A-OPI were sequentially applied by Meriçli and Veloso (2011) for improving the walking stability of a Nao humanoid robot.

The COACH Framework Similarly to A-OPI, when actions are increased/decreased, the COrrective Advice Communicated by Humans (COACHc)[2] (Celemin and Ruiz-del-Solar, 2019) framework employs binary feedback to indicate, for a given state, the direction in which the action taken by an agent has to change, while the magnitude of the change is set as a predefined parameter in the range of the actions. A parametrized policy is directly learned in the parameter space, as in policy search RL (Deisenroth *et al.*, 2013). Differently from A-OPI, the feedback provided in COACHc and the policy updates occur while the agent is interacting with the environment, i.e., during policy execution time, which allows the teacher to directly observe the effects of the corrections and correct again if required, speeding up the learning process. COACHc was originally formulated to model the policy as a linear combination of basis functions, which allowed to solve tasks such as teaching a NAO robot to dribble a ball (Celemin and Ruiz-del-Solar, 2019).

The method was later extended to Deep COACH (D-COACH) (Pérez-Dattari *et al.*, 2018; Pérez-Dattari *et al.*, 2019), which models the policy with Deep NNs, allowing to solve tasks with high dimensional observations, like RGB images obtained from a camera such as in the problem of driving a Duckie Town car (Pérez-Dattari *et al.*, 2019). Also solving problems like balancing a real inverted pendulum swing-up, or

[2]This is different to the other aforementioned COACH in Section 3.1.1, Convergent Actor-Critic by Humans (COACHe), which uses evaluative feedback.

solving a manipulation task in a conveyor belt with partial observations by incorporating memory into the NN architecture (Pérez-Dattari *et al.*, 2020). Furthermore, COACHc was combined with Policy Search RL to learn precise motor skills, solving tasks such as the ball-in-a-cup (Celemin *et al.*, 2019a). These works present experiments in which the learning agents obtained policies with higher performances with respect to the capabilities of their human teachers, who were not always able to execute the task at hand, but still managed to teach it.

COACHc is employed to learn tasks with feedback in the action space, however, corrective advice can similarly be applied to collect feedback in the state space, in tasks wherein the teacher considers that it could be more natural due to the not-so-intuitive relation or effect between the action, the current state, and the next state. With Teaching Imitative Policies in State-space (TIPS) (Jauhri *et al.*, 2021), relative corrections in the state space are used for updating the policy; however, in order to find the action labels that would obtain the advised relative state correction, an additional module based on learning an inverse dynamics model is proposed. This inverse model works for translating the state space feedback into the space of the actions, such that the policy could be updated just as with COACHc. TIPS can also be considered as the interactive version of Behavioral Cloning from Observations (BCO) (Torabi *et al.*, 2018). The method was validated with a *fishing* and a *laser drawing* task with a real KUKA LBR iiwa 7 robot, and a user study with simulated environments showed that using feedback in the state space can reduce the task load of the teachers.

Physical Advice Some works that are more focused on teaching behaviors with manipulators have been proposed for letting the teachers provide kinesthetic corrections over the executed trajectories. These relative corrections are used for either updating a policy or updating the objective function that can be used in a model-based setting with a planner system.

For instance, a policy correction by the teacher on the end-effector displacement with respect to the original trajectory is detected with tactile sensors in Tactile Policy Correction (TPC) (Argall *et al.*, 2011b). The correction could be used for policy refinement or policy reuse. In the

former, the corrections are added as new data points to the training set, whereas in the latter the corrections are used to replace some already existing data points. In both cases, all the data points in the set are used for re-deriving the policy after the execution. The approach was validated with grasping tasks using an iCub humanoid robot.

Additionally, incremental refinement of trajectories of context-dependent policies are performed with kinesthetic feedback in Ewerton *et al.* (2016). The corrections are not detected and computed with tactile sensors, but rather with the measured position difference between the desired trajectory and the one disturbed by the teacher. A reaching task is used in the experiments for testing the method with a BioRob arm. In Canal *et al.* (2016) kinesthetic corrections are also used to reshape a movement primitive used for a feeding assistance robot application.

Relative Corrections as Implicit Preferences The relative corrections intended to modify a manipulator trajectory are also used as implicit preference feedback, despite it being an explicit relative correction in the state space. The trajectory disturbed by the teacher is closer to what the teacher is expecting the robot to do (preferred option) than the trajectory intended by the robot. Some methods leverage this information of preference for learning a function that approximates the teacher's objective (see Equation 2.2), such that it could be used along with a lower-level system for computing the desired robot trajectory. Based on this concept, Trajectory Preference Perceptron (TPP) was proposed and tested in robotic tasks in a household setting and pick-and-place tasks in a grocery store checkout setting (Jain *et al.*, 2013; Jain *et al.*, 2015). Similarly, Online Learning from physical HRI was validated in household tasks with shared workspaces (Bajcsy *et al.*, 2017; Bajcsy *et al.*, 2018; Losey *et al.*, 2022).

Conclusion The methods based on relative corrections allow non-expert teachers to incrementally correct the agent until the unknown desired actions are found, in a guidance setting that resembles the natural way a teacher corrects a student. Some of these methods empower the teachers, who in some cases are not able to demonstrate the task, to teach agents to perform and reach the goals successfully. Learners

outperforming the teacher in IIL is similar to what we see in humans learning complex skills, e.g., when a sports coach guides the player to perform complex behaviors that they cannot do (anymore). Nevertheless, learning with this feedback modality is limited to continuous action problems.

Since this feedback is directly given in either the state or action space, methods using it are also more flexible for reverting the effect of mistaken corrections. Moreover, there are some methods that update the policies with stochastic gradient descent and do not store the feedback in a dataset, which are even less sensitive to the occasional teacher mistakes, allowing to provide a correct label that is not conflicting with any previously stored wrong feedback.

3.3 Discussion

In this section we classified different IIL methods according to the explicit information that is given by the teacher to the learner, having two main categories: Feedback in the evaluative space, and feedback in the transition space. They are divided into subgroups of relative and absolute feedback, therefore, the discussion sets any form of interaction within one of the four subgroups: i) Human reinforcements; ii) human preferences; iii) corrective demonstrations; iv) Relative corrections. Each of the introduced subgroups has its pros and cons which condition the situations in which they could be applied. In general, all these interaction modalities let the teachers train agents that perform better than policies obtained with standard IL, especially to reduce the problem of compound errors, since more complete data is incrementally collected with the teacher interventions during or after the policy roll-outs.

Some works have compared methods of different modalities of interaction and have found that users tend to prefer to interact with the learning agents by communicating information that explains or shows how to perform the task, than to provide assessments of the quality of the policy (Thomaz, Breazeal, *et al.*, 2006; Toris *et al.*, 2012; Amershi *et al.*, 2014; Koppol *et al.*, 2021). However, this preference is not the only relevant factor that could be considered for selecting the most convenient approach to solve a specific problem. In this section, we only

approach that factor and leave the others for the next sections. The rationale for choosing a method should include the answers to questions like *what kind of information is extracted from the feedback?* (Section 4), *is the dynamics model known?* (Sections 4, and 5), *what kind of prior knowledge is available?* (Sections 4, and 6), *is there access to a reward function?* (Sections 8), *what are the independent variables or observations of the policy?* (Sections 6), *what kind of technology is at hand for human-robot communication?* (Sections 9), among others.

The growing community of learning with humans in the loop research is still mostly focused on exploring new methods and evaluating their effectiveness and efficiency. Research for measuring and comparing usability will help to identify what approach or method is more convenient for each kind of problem (See Section 10). Usability can be assessed by analyzing how effective is a method for obtaining a successful policy, how efficient is the learning process, how pleasant the process is for the users, how sensitive it is to human mistakes, and how easy it is for the user to learn to interact with the system.

Nevertheless, there are insights that can guide the selection of the interaction modality to be used for training a policy. Depending on the used modality of interaction, the set of people who can teach a learning agent can be more or less inclusive regarding their expertise. This is correlated with the amount of information contained within the feedback signals of each modality.

With corrective demonstrations, the feedback is the informatively richest since the teacher explicitly shows what the agent should do. This means, that only users with high expertise in the task can teach the system. With the relative corrections, the set of users can be widened because not only expert demonstrators can participate, but also users are enabled to teach if they just have insights about how the transitions would change with a variation of the action. They can advise slight corrections to the agent to incrementally improve a policy. The set of possible teachers is augmented if using human reinforcements, because then, the teachers do not require to know much about what actions should be done or how transitions should be modified, as long as they can assess locally whether each part of a behavior is appropriate (assessments that implicitly happen before any intervention with the two subgroups

of the transition space feedback modalities). If an action is considered wrong, the teacher does not need to know which the correct one is, he/she would just punish it for the agent to try something else until it finds the appropriate behavior. And finally, in the case of learning from preferences, the set of possible teachers is the largest one, since it includes any person who understands the objective of a task, and who can assess whether one behavior is closer to the solution than other ones, without being required to understand or specify what exactly makes the preferred behavior better.

As mentioned before, the corrective demonstrations are the most informative interactions, followed by the relative corrections that are defined in the same domain of actions or states, but they do not need to be strictly accurate since the accumulation of many corrections can gradually reach the desired behavior (Figure 3.5). With human reinforcements, the feedback is a scalar evaluating the performance of each part of the policy execution, and it can be discrete or continuous. With human preferences, the feedback contains the least amount of information because even one discrete feedback signal (or N in the case of rankings) is used to compare full or partial trajectories, without assessing any individual decision.

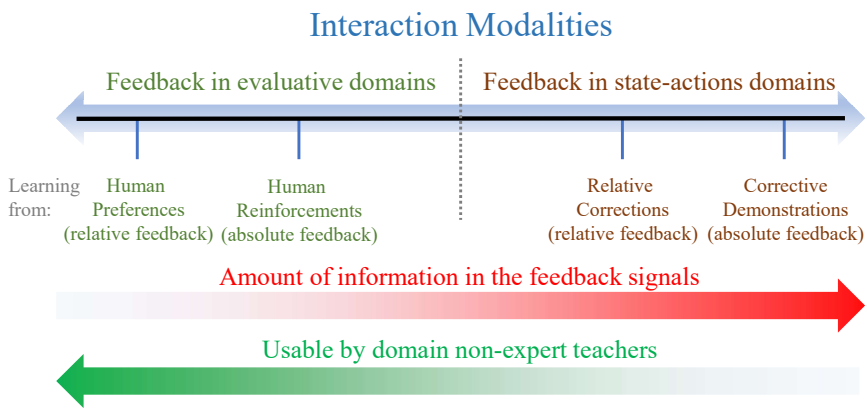


Figure 3.5: Interaction modalities and the information contained in the feedback signals. The four modalities are organized in the plane from the right with the Corrective demonstrations modality requiring the highest expertise, to the left with the human preferences requiring the least (as shown by the green arrow). This order has a negative correlation with the amount of information shared within the feedback signals in each modality (red arrow)

Table 3.2: Summary of IIL methods using feedback in the evaluative and transition domain discussed in this section.

Feedback		List of Papers
Transition	**Absolute**	Thomaz and Breazeal (2007b), Chernova and Veloso (2009), Meriçli *et al.* (2010), Mericli (2011), Ross *et al.* (2011), Chu and Thomaz (2015), Canal *et al.* (2016), Chu *et al.* (2016), Fitzgerald *et al.* (2016), Laskey *et al.* (2016), Schroecker *et al.* (2016), Zhang and Cho (2016), Krening *et al.* (2017), Laskey *et al.* (2017a), Maeda *et al.* (2017), Sun *et al.* (2017), Cheng *et al.* (2018), Fitzgerald *et al.* (2018), Goecks *et al.* (2019), Kelly *et al.* (2019), Menda *et al.* (2019), Ablett *et al.* (2020), DelPreto *et al.* (2020), Mandlekar *et al.* (2020), Prakash *et al.* (2020), Spencer *et al.* (2020), Hoque *et al.* (2021), Bobu and Peng (2022), Chisari *et al.* (2022), Hoque *et al.* (2022)
	Relative	Argall *et al.* (2008), Argall *et al.* (2008), Argall *et al.* (2011b), Argall *et al.* (2011a), Meriçli and Veloso (2011), Meriçli *et al.* (2011), Celemin and Ruiz-del-Solar (2015), Jain *et al.* (2015), Ewerton *et al.* (2016), Bajcsy *et al.* (2017), Bajcsy *et al.* (2018), Pérez-Dattari *et al.* (2018), Celemin and Ruiz-del-Solar (2019), Celemin *et al.* (2019a), Pérez-Dattari *et al.* (2019), Pérez-Dattari *et al.* (2020), Jauhri *et al.* (2021), Luo *et al.* (2021), Losey *et al.* (2022)
Evaluative	**Absolute**	Blumberg *et al.* (2002), Kaplan *et al.* (2002), Thomaz, Breazeal, *et al.* (2006), Thomaz and Breazeal (2007a), Knox and Stone (2008), Mitsunaga *et al.* (2008), Knox and Stone (2009), Knox and Stone (2010), Tenorio-Gonzalez *et al.* (2010), León *et al.* (2011), Pilarski *et al.* (2011), Suay and Chernova (2011), Knox *et al.* (2012), Knox and Stone (2012), Vien and Ertel (2012), Griffith *et al.* (2013), Knox and Stone (2013), Knox *et al.* (2013), Vien *et al.* (2013), Loftin *et al.* (2014), MacGlashan *et al.* (2014), Yanik *et al.* (2014), Knox and Stone (2015), Loftin *et al.* (2016), Krening *et al.* (2017), MacGlashan *et al.* (2017), Arakawa *et al.* (2018), Warnell *et al.* (2018), Arumugam *et al.* (2019), Xiao *et al.* (2020), Kahn *et al.* (2021), Chisari *et al.* (2022)
	Relative	Lund *et al.* (1998), Akrour *et al.* (2011), Akrour *et al.* (2012), Fürnkranz *et al.* (2012), Wilson *et al.* (2012), Jain *et al.* (2013), Akrour *et al.* (2014), Daniel *et al.* (2015), Christiano *et al.* (2017), Sadigh *et al.* (2017), Biyik and Sadigh (2018), Ibarz *et al.* (2018), Palan *et al.* (2019), Bıyık *et al.* (2020), Biyik *et al.* (2020), Brown *et al.* (2020), Myers *et al.* (2022)

Both the limitations given by human factors, or physical constraints like the ones related to learning with real physical robots that cannot be accelerated as in simulation, cannot be directly approached by adding computational power, as in the case of other ML methods. Therefore, some variables like the level of expertise of the teacher, the physical constraints given by the environment and the users, e.g, time, and the available interfaces compose the factors considered for selecting the right modality. Other variables of the interactive imitation learning problem that are discussed in the next sections consider additional nuances of the approach selection. Table 3.2 provides a summary of the methods analyzed in this section.

4

Behavior Representations Learned from Interactions

Section 3 reviewed the different types of feedback that humans can use to transfer their knowledge about a task to robots. This knowledge is ultimately represented with a model encoding implicitly or explicitly the behavior mapping from state/observations to actions. To derive a behavior, different models/representations can be learned from human feedback; therefore, when solving a problem using IIL, it is not only necessary to decide which type of feedback is the most suitable for the task at hand, but also to select the representation that best fits the problem.

In this section, we analyze three groups of models that have been employed in the IIL literature to solve decision-making problems by means of human feedback. The first type of model corresponds to the case where a policy π is directly learned, i.e., a mapping from states to actions is obtained $\pi(s_t) = a_t$.

The second group of models corresponds to state transition learning. In this case, the learned model π^s represents a desired state transition (or its derivative) as a function of the current robot's state $\pi^s(s_t) = s^d_{t+1}$. These methods depend on another module in charge of executing the desired state transition, e.g., a feedback controller or an inverse dynamics model $\mathcal{T}^{-1}(s_t, s^d_{t+1}) = a_t$.

The third group consists of implicit modeling of behaviors through cost/reward/scoring or value/utility, which can be then optimized for finding the corresponding actions or transitions.

4.1 Direct Policy Learning (Actions)

As stated in the introduction of this section, Direct Policy Learning (DPL) stands for those methods where a policy $\pi(s_t) = a_t$ is obtained as a result of the IIL process. From the literature, it is possible to observe that methods apply DPL by means of different feedback modalities (see Section 3), ranging from cases where desired actions are explicitly indicated to the robots, to other cases where direct policies are learned through action evaluation.

4.1.1 Intuitive Action Spaces

We start by considering methods that employ corrective feedback for DPL. In such cases, it is necessary to indicate to the robot how to modify its actions to improve its behavior. Therefore, in cases where the action space of a robot is high dimensional (e.g., bimanual task) or unintuitive (e.g., underactuated hand (Della Santina *et al.*, 2018)), it can be very challenging for a teacher to provide corrections to the robot. Consequently, DPL with corrective feedback is limited to those cases where the action space is intuitive for humans. An example of this is the problem of autonomous driving consisting on three actions: *throttle*, *steering angle* and *brake*. Given that many humans know how to drive a car, it is also intuitive for them to teach a robot how to control these actions appropriately.

In these cases, absolute corrections (demonstrations) provide a well-suited framework for DPL, since the human feedback directly indicates to the robot the desired action for a given state. Several DAgger-based methods (see Section 2.2.5) have attempted to solve robotic tasks by learning policies directly. However, in many cases, they were only validated in simulated scenarios with simulated teachers (Prakash *et al.*, 2020; Zhang and Cho, 2016; Menda *et al.*, 2019; Hoque *et al.*, 2021).

As explained in Section 3.2.1, Dagger-based approaches are limited when learning from humans, as the labeling process is unintuitive and prone to errors; however, the subset of these methods based on human interventions can alleviate this issue. Consequently, methods based on human-gated (Kelly *et al.*, 2019; Goecks *et al.*, 2019; Mandlekar *et al.*, 2020; Luo *et al.*, 2021) and robot-gated (Hoque *et al.*, 2022; Laskey *et al.*, 2016; Ablett *et al.*, 2020) interventions have been successfully applied to learn these policies with human teachers, solving autonomous driving and manipulation problems. It is worth noting that, although learning from interventions has been proposed in recent years as a variation of DAgger for DPL, similar strategies were already being proposed more than a decade ago (Chernova and Veloso, 2009).

4.1.2 Intuitive, Yet Challenging Action Spaces

The methods discussed above have proven to be an effective tool for DPL; however, there are some cases where the action space of a robot might be intuitive, but challenging to demonstrate. High-frequency tasks are a good example of this (e.g., swing-up pendulum), where, although the dynamics and control inputs of the robotic system can be well-understood by the teacher, it can still be challenging for them to successfully control the robot and provide absolute corrections.

In such cases, relative corrections are a suitable alternative for DPL, since the teacher only needs to indicate the direction where the action taken by the robot must be modified. This allows to gradually improve the behavior of the learner even when it is not possible to demonstrate the task. For instance, Celemin and Ruiz-del-Solar (2019) train an agent to balance a bicycle and solve the cart-pole problem through relative corrections, where it is shown that, even though the human is not able to teleoperate both tasks, it is capable of successfully teaching an agent to solve them. Moreover, one of the deep learning extensions of this method (Pérez-Dattari *et al.*, 2020) was able to successfully teach an agent to swing up a pendulum from raw pixels of an image, showing superior performance of relative corrections in this task when compared to the intervention-based method HG-DAgger (Kelly *et al.*, 2019).

4.1.3 Non-intuitive Action Spaces

Some approaches have addressed the problem of DPL in scenarios where the action space can be difficult for the teacher to understand. For instance, giving corrections in the joint space of a robot arm corresponds to one of these cases. Providing corrections in this space can be very challenging, since, commonly, manipulators must solve tasks by controlling their end-effector. Hence, the teacher needs to have an accurate internal model of the effect that actions in joint space generate in the robot's end-effector to teach the robot how to solve the task at hand (Section 4.2 provides a deeper study of these cases). To address this problem, Jauhri *et al.* (2021) introduce a method for giving relative corrections in the state space of the environment (e.g., feedback about the effects of the manipulator's end-effector) and use an inverse transition model (learned from interactions with the environment) to map this feedback to the action space (joint level in the example) of the robot, allowing it to directly learn a policy in this space.

Finally, evaluative feedback can be employed to learn policies directly when the action space is not intuitive for the teacher. Evaluative feedback allows the human to teach a policy without having to understand the action space of the robot (see Section 3.1). In most cases, evaluative feedback is employed to learn a value function. Nevertheless, MacGlashan *et al.* (2017) propose to directly learn a policy through evaluative feedback by introducing an IIL method inspired by policy gradients from RL (Sutton and Barto, 2018). The deep learning extension of this method (Arumugam *et al.*, 2019) showcases experiments where an agent learns to solve simulated navigation tasks from raw pixels of an image.

As a final remark, note that methods that work in non-intuitive action spaces can also be employed to learn policies in intuitive action spaces.

4.2 Learning Desired State Transition/Dynamics

Desired State Transition Learning (DSTL) stands for those methods where a policy indicates the next desired state that the agent should

visit $\pi^s(s_t) = s^d_{t+1}$. As highlighted in Section 4.1, when it is not intuitive to provide feedback in the action space of the agent, solutions have been proposed for converting human feedback in state space to the desired action space. For example, in a torque-controlled robot, the demonstrator's action sequence is not available/observable when learning from kinesthetic teaching (Celemin *et al.*, 2019a). This is also true when humans want to imitate each other's behavior: they do not have access to the internal actions used by the demonstrator. Instead, they can only observe the state transitions generated by those demonstrations; therefore, they need to infer the necessary actions to achieve the same transitions. Additionally, when dealing with robots with many Degrees of Freedom (DoF), the user aims to teach behaviors in task space, and not in the actuator level of the desired task to perform. For example, in a manipulation insertion task, the demonstrator would only focus on the end-effector position and not on the complete kinematic chain state of the robot. Moreover, teaching and correcting movements in task space also allow better generalization (Ewerton *et al.*, 2016; Mészáros *et al.*, 2022).

DSTL relates to the literature of *model-based* methods in that it makes use of learned/known models of the environment to achieve desired state transitions. The IIL community aims to leverage those models to make the teaching and the correction easier and accessible to non-expert users, who may not be familiar with the effect of actions on the operational space dynamics (Argall *et al.*, 2011a). For example, manipulation tasks can be solved by interactively learning the desired end-effector transitions. Those transitions can then be achieved by the robot using a feedback controller, such as cartesian impedance (Franzese *et al.*, 2021b) or velocity (Chisari *et al.*, 2022) controllers.

State Dependent vs State Independent Transitions The desired state transitions can be described as a function of the current state or with a state-independent formulation; for example, as a function of time/phase using movement primitives like Dynamic Movement Primitive (DMP) or Probabilistic Movement Primitive (ProMP) or as a function of the current state space using a Gaussian Process (GP) (Franzese *et al.*, 2021b) or a NN (Chisari *et al.*, 2022) (introduced in Section 6). When

learning a state-dependent transition the policy can be described as $\pi^s(s_t) = s^d_{t+1}$ that still fits the MDP formulation. On the other hand, when learning a state-independent movement primitive, the robot learns a behavior model that describes how the state evolves over time, i.e. $\pi^t(t) = s^d_{t+1}$, which can again be tracked using feedback control. Argall *et al.* (2011a) use human feedback to shape the desired trajectory of a robot starting from the assembly of primitives and using local correction refinement of the final driving policy execution. Celemin *et al.* (2019a) combine interactive learning with RL to learn a DMP. The interactive correction on the current robot state is used as exploration for a faster convergence to optimal performance of the task. Similarly, Schroecker *et al.* (2016) use IIL to stop the RL algorithm, and they let the user bring the robot to the desired state at the particular time of the execution. This strategy is used to create soft via-point constraints that limit the RL exploration and results in a faster learning convergence of a DMP. An active and interactive collection of state space demonstrations is proposed by Maeda *et al.* (2017) where multiple trajectories are collected from demonstration, and saved in a model that estimates the movement parameters as a function of the current context of the task. When the robot faces a novel situation, it actively queries the user to provide more data.

As a final remark, the difference between desired state transition and action strongly depends on the definition of the MDP: if the action is set to be equal to the desired robot transition, the two categorizations collapse in a single one. However, when working with real robots, the re-formulation of the MDP needs to be supported by available models (learned or hard-coded) that handle the conversion from the newly defined action and the actual robot control. This section summarized how interactive methods took advantage of these models to obtain more efficient and user-friendly teaching of robotics tasks.

4.3 Learning Reward and Objective Functions

Contrary to learning a policy from human feedback directly, there exists a plethora of methods that fit a reward or objective function first. Such a function can then be used to either derive a policy (Christiano *et al.*,

2017) or directly infer the optimal action (Knox and Stone, 2008). In the next subsections, we describe and distinguish between methods that learn a reward, cost or scoring function, and methods that learn an objective, such as value or utility functions.

4.3.1 Learning Reward, Cost, or Scoring Functions

A common approach studied in the literature is to use human feedback to shape a *reward* function, which can then be used to train a policy. For example, in the method proposed by Christiano *et al.* (2017), the reward function is estimated by minimizing a cross-entropy loss where the labels are the human-provided pairwise preferences. An alternative approach is taken by Sadigh *et al.* (2017), where the weights of a reward function are learned from human preferences via a Bayesian update. An extension of these ideas consists of learning a reward function by integrating human preferences with expert demonstrations. For example, in DemPref (Palan *et al.*, 2019; Biyik *et al.*, 2020), the weights of the reward function are pretrained with the expert demonstrations via a Bayesian IRL approach, while Ibarz *et al.* (2018) use the demonstrations to pretrain the policy, which is then used to generate a set of initial trajectories for the human teacher to annotate with preferences, from which the reward function is finally learned.

While the aforementioned methods aim to learn a reward function via preference feedback, there have been works in the literature proposing to learn approximate reward functions from corrective feedback. In the works of Bajcsy *et al.* (2018) and Losey *et al.* (2022), the robot is trained through physical human-robot interaction, and the weights of the reward function are updated in the direction of the Maximum a Posteriori (MAP) estimate of the observed and corrected trajectories. A different approach is taken in LaND (Kahn *et al.*, 2021), where human intervention is used to learn a disengagement prediction model to distinguish good and bad actions. This model is then included in the *cost* function used to guide a model predictive control planner.

A different solution is TPP (Jain *et al.*, 2013; Jain *et al.*, 2015), which consists of learning a *scoring function* used to rank different trajectories. At each episode, a planner is used to sample multiple trajectories, which

are ranked according to the current scoring function. If the top scoring trajectory is not considered adequate, the human teacher provides a preferred alternative, which is used to update the weights of the scoring function by gradient-free optimization. At inference time, the scoring function is used to pick the best trajectory from the ones sampled by the planner.

4.3.2 Learning Value or Utility Functions

Instead of learning a reward function, an alternative approach consists of directly training from human feedback a *Q-value* function, which describes the expected future cumulative reward of each state-action pair (Watkins and Dayan, 1992). As shown by Thomaz and Breazeal (2006), Thomaz, Breazeal, *et al.* (2006), and Suay and Chernova (2011), this is achieved by substituting the environment-provided reward (which is not always readily available) with the human-provided evaluative feedback. In these works the action space is discrete; hence, the agent can infer the action by directly maximising the Q-function by selecting the action with the highest value: $a = \arg\max_{\hat{a}} Q(s, \hat{a})$. A similar approach is taken in the TAMER framework (Knox and Stone, 2009), discussed in Section 3.1.1, where the human reinforcement function $H(s, a)$ is trained from human feedback and then used directly for inference: $a = \arg\max_{\hat{a}} H(s, \hat{a})$. Directly maximizing the value function at inference time is an effective approach, but it is only straightforward for discrete action spaces.

A possible solution to cope with continuous action spaces is to use the learned objective function for learning an explicit model of a policy. PPL (Akrour *et al.*, 2011; Akrour *et al.*, 2012; Akrour *et al.*, 2014) uses the preference feedback from the teacher to learn a policy return estimate, which is then used to build new candidate policies. The policy return estimate, also referred to as *utility function*, represents the quality of a given state with respect to discovering new better policies. New policies are then generated following either an evolution strategy approach or an Expected Utility Selection (AEUS) criterion, and the process is repeated.

4.4 Discussion

In this section, we classified methods according to the type of model learned from human feedback. Different approaches exist to teach robots to solve tasks through human-robot interactions. Some methods directly learn a mapping from states to desired action/states and others indirectly derive policies by learning functions that evaluate the performance of the agent's actions.

We observed that depending on the task at hand, and how intuitive it is for a human to provide a specific type of feedback, different alternatives exist to address it. Therefore, although one type of feedback can be ideal for a specific use case (e.g., interventions for driving), it might not be a satisfactory candidate to solve other problems (e.g., interventions for balancing tasks).

Furthermore, we also analyzed that the learning process can be assisted by incorporating prior knowledge about robotic platforms in IIL methods. For instance, it is possible to directly learn manipulation tasks in the robot's end effector space by using the robot's geometry to map these behaviors to the actuation space. More details regarding auxiliary models are presented in Section 5.

Finally, we observed that it is possible to employ feedback signals, that require a low cognitive load for human teachers, to implicitly learn behaviors using reward or objective functions. These behaviors can then be decoded using optimization methods such as model predictive control.

So far, we studied the feedback modalities that humans can employ to transfer their knowledge to robots (Section 3) and, in this section, the models that can be learned from them (summarized in Table 4.1). We observed that depending on the model at hand, different types of feedback can be used to learn it. However, an important part of model learning is to select an appropriate function approximator to represent it, which was not addressed in this section. This is reviewed in depth in Section 6.

Table 4.1: Summary of IIL methods discussed in this section.

Behavior Representation	List of Papers
Direct Policy	Chernova and Veloso (2009), Laskey *et al.* (2016), Zhang and Cho (2016), MacGlashan *et al.* (2017), Arumugam *et al.* (2019), Celemin and Ruiz-del-Solar (2019), Goecks *et al.* (2019), Kelly *et al.* (2019), Menda *et al.* (2019), Ablett *et al.* (2020), Mandlekar *et al.* (2020), Pérez-Dattari *et al.* (2020), Prakash *et al.* (2020), Hoque *et al.* (2021), Jauhri *et al.* (2021), Luo *et al.* (2021)
Desired State Transition	Argall *et al.* (2011a), Ewerton *et al.* (2016), Schroecker *et al.* (2016), Maeda *et al.* (2017), Celemin *et al.* (2019a), Franzese *et al.* (2021b), Chisari *et al.* (2022), Mészáros *et al.* (2022)
Reward/ Cost/ Scoring	Akrour *et al.* (2011), Akrour *et al.* (2012), Jain *et al.* (2013), Akrour *et al.* (2014), Jain *et al.* (2015), Christiano *et al.* (2017), Sadigh *et al.* (2017), Bajcsy *et al.* (2018), Ibarz *et al.* (2018), Palan *et al.* (2019), Biyik *et al.* (2020), Kahn *et al.* (2021), Losey *et al.* (2022)
Value/ Utility	Thomaz, Breazeal, *et al.* (2006), Knox and Stone (2008), Knox and Stone (2009), Suay and Chernova (2011), Christiano *et al.* (2017)

5

Auxiliary Models

When teaching autonomous agents interactively, the quality of the human feedback directly affects the learned behavior. Section 4 discussed different types of models/representations that can be used to encode the desired behavior. Besides these, it is often useful to employ additional models to enhance the training process and/or the execution of the task. In this section, we consider multiple types of auxiliary models (see Figure 5.1) that are used to improve the learning process, the teacher experience, and the policy execution with respect to different objectives such as data efficiency, safety, and credit assignment. These models can be either learned or hand-crafted and are applied either at training time or at inference time.

5.1 Task Features Learning

Learning features for a task allows identifying a more compact and/or descriptive state space, and it can tremendously boost learning speed and generalization. As such, learning the importance of features, or the features themselves can be a key component on IIL. This is especially important within the IIL scope, given the necessity of having a reduced number of demonstrations and corrections and still achieving desirable

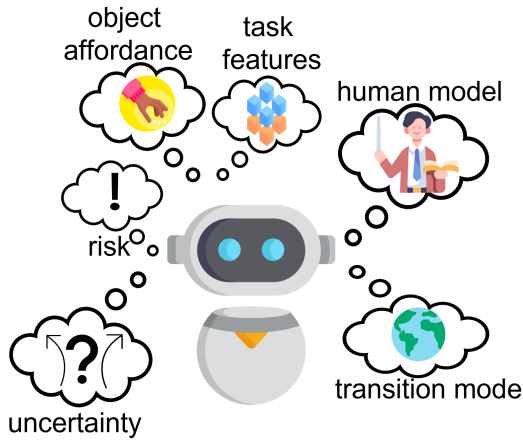


Figure 5.1: Auxiliary models that are commonly used in interactively imitation learning.

performance. Furthermore, hand-designed task features are undesirable as they might not generalize or transfer to different tasks and they require task-specific engineering effort.

Task feature learning has been applied in the IIL context in the past years, aiming for identifying features for improving the learning performance, enabling teaching with high-dimensional signals, or aligning the agent's and the human's knowledge (Bobu and Peng, 2022).

Dimensionality Reduction Dimensionality reduction approaches target finding a compact set of structured and informative features from high-dimensional signals. Such approaches aim to enable IIL methods to work on high-dimensional observations without requiring large amounts of data as is the case with most deep learning methods. Auto-encoders have been successfully applied in order to reduce the high-dimensional state spaces by learning a smaller set of informative features. They have been used in D-TAMER (Warnell *et al.*, 2018), in D-COACH (Pérez-Dattari *et al.*, 2018), and by Luo *et al.* (2021), however, in those works, the auto-encoders are pre-trained, instead of being interactively learned from human input.

In this scope, Pérez-Dattari *et al.* (2019) extend D-COACH by making both the auto-encoder and policy to be learned online. Results show data efficiency improvement w.r.t. the previous version of D-COACH, which is due to the fact that both the auto-encoder and the learner agent are trained using the same distribution, i.e., the states encountered during training, while auto-encoders trained offline are likely to encode only a set of all the possible observations the learner can obtain.

Pérez-Dattari *et al.* (2020) extend these architectures to enable IIL from camera images in environments that are non-fully observable. A Long Short-Term Memory (LSTM) model is used to introduce recursion in the learned NN for dealing with partially-observable processes. The approach is applied to a fruit selection task, in which fruits approach the robot through a conveyor belt and a camera captures a region of the belt different from the robot's action region, making it necessary to learn to predict where the fruits will be and when they will arrive at the robot's reach.

Interactive Feature Learning for Inverse Reinforcement Learning Relevant task features can also be learned interactively within a IRL setting. Bobu *et al.* (2021) and Bobu *et al.* (2022) present Feature Expansive Reward Learning (FERL), a framework that interactively learns arbitrarily complex non-linear features, such as distance to and between objects. Then, these features are used to obtain a policy via IRL. At the beginning of the first step, the robot does not have prior knowledge about the features, and the user provides information about them through the so-called *feature traces*. This feedback consists of trajectories obtained from the user, who guides the robot from states where the features are strongly activated to states where they are not, according to the teacher's judgment. In the second step, the learned features are combined using standard reward learning frameworks, which are used to infer the policy. Results show that the learned features improve generalization capabilities w.r.t. non-interactive IRL-based approaches (Finn *et al.*, 2016; Wulfmeier *et al.*, 2016), which simultaneously learn task features and reward.

5.2 Object Affordances

Many real-world robotic applications require finding a sequence of actions applied to relevant objects to bring the environment to an intended state. To be effective, sometimes it is useful to design appropriate abstractions of the environment, to focus only on the actions that generate the intended (meaningful) effect on the objects. This can be achieved via affordance modeling. An affordance model accounts for a high-level behavior modeling approach that learns the relationships between the robot, its actions, and their effect on objects (Gibson, 1977). For a given object, an affordance $\mathcal{AF}_{\mathcal{I}}$ represents a subset of the state-action space ($\mathcal{AF}_{\mathcal{I}} \subset \mathcal{S} \times \mathcal{A}$), that leads to the intended effect $\mathcal{I}$ (mapping from states to distribution over states, denoting high-level effect, e.g. *grasped*) (Khetarpal *et al.*, 2020). There are different approaches to discovering meaningful affordances as described later in this section. Once discovered, affordances might offer a kind of generalization across different objects of the same class. High-level decision-making can utilize then affordances to achieve desired goals/transitions in the environment efficiently and effectively.

Learning Object Affordance from Vision Vision has been used since the beginning of affordance learning in the literature. Thomaz and Breazeal (2007b) and Thomaz and Cakmak (2009) address human-guided exploration for learning affordances. In this approach, the robot learns object affordances through a combination of self-exploration and human guidance. The teacher guides the robot by i) providing evaluative feedback, ii) suggesting to perform certain actions (e.g., *try action X on the object Y*), iii) drawing attention to an action-effect observation (e.g., *look it's Z*) and iv) controlling the environment so that the appropriate cues are most noticeable for making the learning process efficient (placing an object in areas or poses which increased the likelihood of finding affordances).

For instance, Thomaz and Cakmak (2009) experiment with a robot configured as an upper torso humanoid. It learns about a set of five objects with different geometrical shapes and colors. The robot learns different affordances, such as *lift-able*, *open-able*, *roll-able*, *move-able*

and *tip-able*. The results from these experiments show that a non-expert user is able to teach the robot to learn the affordances more successfully than when the robot learns by itself.

Although human guidance results in efficient exploration strategies, it is cumbersome to have humans provide an exhaustive set of interactions for each affordance. Therefore, to reduce the burden on humans, Chu and Thomaz (2015) introduce the *human seeded exploration* strategy, which is a combination of self-exploration and human-guided exploration. In this strategy, first, a human teaches the affordances using kinesthetic demonstrations. The robot uses the distribution of the demonstrations for searching in the action space and finding affordances in new situations. The experiments were conducted with a robot having 7 DoF arms. This approach results in improved success rates with fewer object interactions for learning affordances.

Learning Object Affordance from Haptic Feedback The previously mentioned approaches consider learning affordance models using visual information. In contrast, Chu *et al.* (2016) propose a complementary haptics affordance model, which is fused with visual affordance to develop a multi-modal model. It characterizes how a particular action-object pair feels and, therefore, can aid in better task completion. During each interaction, the human teaches the robot via *environmental scaffolding*, i.e., moving objects slightly to perturb the action context.

Learning Object Mapping Using Affordance Fitzgerald *et al.* (2016) and Fitzgerald *et al.* (2018) develop an IIL method to enable agents to map representations of objects between environments based on their affordances. These models are used to transfer a learned task into a new environment. For example, a glass (*fill-able*, *drink-able*, *pour-able*) in an environment E_a can be mapped to a pitcher (*fill-able*, *pour-able*) in another environment E_b, for instance, on the basis that both objects are *pour-able* if the task requires pouring. However, these objects could not be mapped together if the task requires drinking since the pitcher is not drinkable. The human teacher assists the agent by indicating the correct mapping for some of the objects. Through this assistance, the algorithm infers the mapping function between source and target

objects, such that the remaining objects can be mapped autonomously for task completion. The results show that the agent can use human guidance to quickly infer a correct object mapping, requiring assistance with only a few steps.

Learning Affordance from Natural Language Advice In order to learn a task from users with non-ML expertise, an IIL method should be able to understand simple human explanations. Most approaches require the users to provide state-specific advice, which might not always be intuitive for them. Alternatively, it is possible to provide object-specific advice through natural language (Krening *et al.*, 2017). For example, consider that a human wants to teach an agent to play a Super Mario game. It is more intuitive for the human to advise which actions to take with respect to an object (e.g., *jump on an enemy*), rather than state-specific advice such as *hold the jump key for 10 frames when Mario is within 2.5 horizontal blocks of an enemy with a velocity of 3.2 units/frame*. Hence, object-focused feedback helps generalize over the state space.

5.3 Forward and Inverse Transition Models

Transition models can be useful for training autonomous agents as they encode information about how the state evolves given a certain action, and can potentially result in better sample efficiency and improved generalization capabilities.

A useful application of a transition model consists of the search for an optimal trajectory given a cost function. In the setting of interactive learning, Losey *et al.* (2022) use the optimization-based motion planner TrajOpt (Schulman *et al.*, 2014) to find the best new trajectory according to a reward function shaped by human corrections. Similarly, various methods that learn a reward function by preference feedback make use of the dynamics model of the system to optimize the output trajectory (Sadigh *et al.*, 2017; Palan *et al.*, 2019).

Another use of transition models is trajectory generation for sampling-based approaches. In TPP, Jain *et al.* (2015) employ a model of the robot kinematics and of the obstacles in the environment within

a sampling-based planner to generate collision-free trajectories. The learned *scoring function* obtained from the human feedback is used to pick the optimal path. A further sampling-based approach is employed in LaND (Kahn *et al.*, 2021), where at each step multiple roll-outs are generated, and the first action of the best trajectory is applied to the agent in a model predictive control fashion. The cost function for the selection includes a model, trained from human feedback, which outputs a sequence of disengagement probabilities, i.e. the probability that the robot needs to be stopped by the human teacher because it moves towards an unsafe region of the state space.

In the field of IIL, another application of transition models is to facilitate the teaching process, making it more intuitive for non-expert demonstrators that are not familiar with the effect of controlling robot actions. The TIPS framework of Jauhri *et al.* (2021) proposes to learn a forward dynamics model which maps a state and action pair (s_t, a_t) to the next state s_{t+1}. This forward model is continuously updated from the recorded robot transitions during the task's episodes and used to allow the user to give feedback on the desired state dynamics rather than the desired action. This approach is advantageous for all applications where providing feedback in the action space is difficult or unintuitive.

5.4 Confidence, Novelty and Risk Models

When learning from non-expert users, it is desirable to minimize the amount of feedback the teacher needs to provide. At the same time, the robot should not attempt dangerous actions in regions where the learned policy is not well trained. To achieve these objectives, different researchers use confidence, novelty and risk detection models to support the user during the teaching.

5.4.1 Novelty and Confidence Model

The use of Bayesian methods (and their approximations), through the estimation of epistemic uncertainty, provides a successful tool for performing safe learning, i.e., calling attention when in a region of high uncertainty or asking explicitly for more data (active learning). Chernova

and Veloso (2009) propose a confidence-based interactive method where the policy outputs the action as well as its confidence. When the learner executes the action, it also checks its confidence score against a threshold. If below, the user is asked to intervene. Similarly, Franzese *et al.* (2021b) encode the desired robot motion from demonstrations and interventions, and use a measure of confidence to constantly direct the robot to regions of minimum uncertainty (high confidence).

To enhance exploration when learning from interaction, Kulak *et al.* (2021) adopt a measure of confidence to explicitly bring the robot to regions of high uncertainty and actively ask for feedback to have a heterogeneous set of demonstrations to learn from. Menda *et al.* (2019) and Kelly *et al.* (2019) propose to derive the action and confidence estimate as the mean and variance of the predictions of an ensemble of NNs. The threshold over which the user's help is queried is tuned with previously observed interventions of the user. Subramanian *et al.* (2016) estimate a value function and use two statistical measures (leverage and discrepancy) to compute the influence of visited states on the value estimation. When it is high, the teacher is actively requested to perform a demonstration to move the robot to that state to increase exploration. In the context of Bayesian IRL, Cui and Niekum (2018) propose an active and interactive framework. The agent queries the user for segmenting and labeling (segment of) trajectories as good or bad. Given the probabilistic formulation of the reward function, the robot can generate trajectories that maximize information gained after the actual labeling from the user. Finally, the user can label different steps as good or bad, and this label is then used to update the reward function using a soft-max update rule.

5.4.2 Risk Detection and Safety Enhancement

While very important, the novelty of the observation is not the only factor that leads to a high probability of failure. A state can be risky even though not novel. In the context of IRL, to generate risk-aware performance-based trajectories, Brown *et al.* (2018) examine the robot's policy and evaluate per-state policy loss. The states that have high loss/low cumulative reward are classified as risky states. Starting from

these critical states, the algorithm samples trajectories and asks the human supervisor to critique them. With human feedback, the policy is updated to reduce the risk of failing when approaching those states.

Zhang and Cho (2016) propose SafeDAgger, a safety rule to enhance DAgger, where during the data collection, not only the action policy is learned, but also a safety policy that returns a binary classification value, i.e. safe/unsafe, given the state-action pair from the primary policy. LazyDAgger (Hoque *et al.*, 2021) extends this simple rule with a hysteresis model to reduce the amount of switching between the supervisor and the policy. Similarly, Ablett *et al.* (2020) propose to fit a discriminator that is able to classify state-action pairs as dangerous and assign them a probability of failure. In order to avoid querying the user too often, an additional parameter is learned from interaction, which evaluates if the current discrimination is too dependent (high user burden) or too independent (leads to failure), and updates it accordingly. Finally, Hoque *et al.* (2022) propose to learn a Q-function to estimate the probability of failure from a certain state, based on past experience. This measure is used to request the user to intervene.

5.5 Human Models for Feedback Interpretation

When receiving feedback from humans, it is important to take into account the delayed reaction and the decaying significance of past state-action pairs. The authors of TAMER (Knox and Stone, 2009) propose a '*Credit Assigner* module, intended for environments of *high frequency* regarding the human response capabilities. The module aims to solve a temporal credit assignment problem. A human trainer is not able to assess the effect of each action at each time step, so this produces a delay between the action execution and the human response. The Credit Assigner proposed in TAMER approaches this problem by associating the feedback not only to the last state-action pair but to a past window of pairs. Each pair is weighted with the corresponding probability computed with a model of the human delay probability density function (Knox and Stone, 2009). While being determined experimentally, this model of the human delay appears to correspond to the one found empirically in psychological studies (Sridharan, 2011).

A similar idea is presented by Loftin *et al.* (2016) with the I-SABL framework. They propose a probabilistic model of the human teacher feedback which describes how a trainer decides to provide an explicit reward or explicit punishment. The model takes into account the overall teaching strategy, and it is also able to learn from the actions for which the human does not provide any feedback. Finally, Celemin and Ruiz-del-Solar (2015) propose to employ a model of the human teacher to predict what feedback the agent will receive on a given state. This model can then be used to adapt the step size of the relative corrections from the teacher, being able to incorporate the intentions of the teacher encoded on the past feedback, i.e., for a specific state, either applying a large change to the policy or fine-tuning it.

5.6 Discussion

In this section, different auxiliary models are discussed that contribute to improving the overall interactive learning process. Task features learning addresses the problem of obtaining a good representation of the environment that a policy can use to learn data efficiently and generalize well. Object affordances are useful for finding appropriate abstractions of the objects to manipulate and to focus the learning process on the effects that certain actions generate. Transition models are helpful, as they describe how the state evolves given a certain action, and are often employed for trajectory generation, either via sampling or optimization. Confidence, novelty and risk model are employed to assess the safety of the agent behavior in a given state, and to react accordingly if needed, e.g. by querying the human teacher. Finally, models of the behavior of the human teacher can also be used. They aim to improve different aspects of the learning process such as credit assignment or adaptive step size in the feedback interpretation. Hence, each type of auxiliary model has its advantages, and the choice of adopting one over the other mainly depends on the task at hand: task features can be useful for problems with otherwise high-dimensional representation, object affordances are mainly tailored for manipulation tasks, transition models are needed for scenarios where planning is required, while confidence and risk models are helpful for safety-critical application. Moreover, such models are not

mutually exclusive, and a combination of multiple of them is possible. In the next section, the existing types of representation or function approximation that are commonly used in the IIL setting are discussed.

6

Model Representations (Function Approximation)

In Section 4, different models learned from interactions are analyzed, namely policies, rewards/objectives, and desired state transitions. These mathematical objects are estimated using finite data but are required to act on a continuous space, e.g. to generalize to previously unseen states, see Figure 6.1. To achieve the desired approximation, depending on the characteristics of the task at hand, design choices must be made on the function model, e.g. linear, non-linear, parametric, non-parametric, etc.

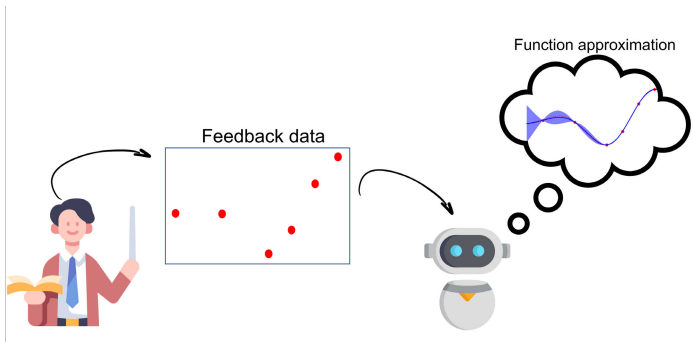


Figure 6.1: Function Approximations: the feedback of the teacher is used by the robot to reconstruct a function.

The appropriate model representation enables, for instance, to cope with small/big databases, noisy/conflicting data, and a continuous stream of new information coming from the environment or from the teacher. This section summarizes regressors and classifiers that have been adopted in the field of IIL. Many of these methods could be considered as a particular case of a unified model (Stulp and Sigaud, 2015), showing the usability of different function approximations to different interactive learning algorithms.

6.1 Linear Models

The simplest example of function approximation is linear regression, where the objective is to map inputs X to a continuous output Y with a linear dependency by means of a parameter vector Θ, i.e,

$$Y = X\Theta. \tag{6.1}$$

6.1.1 Combination of Features

More generally, the function can be obtained with a linear combination of hand-crafted features that have a particular meaning in the context of the studied problem. The prediction is obtained as

$$Y = \Phi(X)\Theta, \tag{6.2}$$

where Φ corresponds to the chosen features.

For example, Spencer *et al.* (2020) obtains the value function as a linear combination of different features, i.e. boundary violation, absolute curvature to the next step, distance to the nearest obstacle, and distance to the goal path. Subramanian *et al.* (2016) use this type of model to represent the value function for solving a task with reinforcement learning. These models are also employed to compute statistical measures that were used to actively query the user when necessary. In the field of reward learning from demonstration and correction (Bajcsy *et al.*, 2017) or preferences (Sadigh *et al.*, 2017), a linear model of hand-crafted features is investigated for solving the POMDP through online inference. This function approximator has the advantage of easily integrating prior knowledge in the model while keeping a linear complexity.

6.1.2 Radial Basis Functions

Radial Basis Function (RBF) correspond to a specific type of feature model that has been widely used in the IIL literature. Its value depends only on the distance between the input and some fixed points. Knox and Stone (2009) use RBFs policies to encode non-linear policies that are updated online from human reward. Similarly, Celemin and Ruiz-del-Solar (2015) also use RBFs controllers where the policy is updated with feedback in the action domain. MacGlashan *et al.* (2017) use an actor-critic strategy to update the policy parameters. In fact, using the same update rule based on gradient descent, the user can provide policy-dependent feedback, i.e., the expected value function of that state, or advantage function, that describes how much better or worse an action selection is compared to the agent's performance following its policy. The RBF models allow fitting smooth functions, ideal for direct robot control with a small set of parameters. The location and the width of the basis functions can be chosen based on prior knowledge. The main advantage of these models is the stable update of their parameters, each new data point has a local effect on the function after the update, something very convenient for incremental learning approaches.

6.1.3 Locally Weighted Regression

One alternative for solving Equation 6.1 is finding the optimal parameters Θ with the pseudo-inverse of X, $\Theta = (X^T X)^{-1} X^T Y$.

This formulation can be extended to fit nonlinear relations by augmenting the parameter space with a weight matrix W, known as Locally Weighed Regression (LWR): $\Theta = (X^T W X)^{-1} X^T W Y$, where W is a symmetric matrix and each element is defined by a function that tells you how much the individual values of X and Y should be considered when fitting a line through the neighborhood of X. Hence, given a database, the prediction can be obtained as a weighted average of all the terms.

LWR has been applied for fitting non-linear functions without the requirement of choosing the degree of the function approximator nor the number of necessary parameters such as in the RBF controller. For

example, Argall *et al.* (2008) and Argall *et al.* (2011a) employ LWR to fit a policy from state-action pairs that are collected from demonstrations or interactively updated using a teacher's advice. When feedback is provided, a new data point is aggregated only if the query input is not close to the original data, avoiding the collection of conflicting or redundant labels in the database.

6.2 Gaussian Process

Gausian Processes (GPs) provide the means for making predictions while incorporating prior knowledge about the distribution of the data (Rasmussen and Williams, 2006). We assume that all training and test labels are drawn from an $(n+m)$-dimensional Gaussian distribution $\mathcal{N}(0, \Sigma)$, where n is the number of training points, m is the number of testing points and Σ corresponds to the covariance matrix. When querying the model on m test inputs, its prediction is an m-dimensional gaussian distribution, with mean and variance, after conditioning the $m+n$ distribution on the n training points. The variance quantification is a measure of the epistemic (model), uncertainty. The statistical formulation and the quantification of uncertainty make the GP desirable when limited data are available; however, on the other hand, they do not scale well with big databases or high-dimensional inputs.

Wout *et al.* (2019) use GPs for computing the uncertainty of the policy used for active queries, for approximating the human model used for adapting the learning rate, and for deciding whether the new data instances interactively obtained are used to be appended in the dataset or to modify a data point of the dataset. GPs have been successfully applied for learning reward functions using human preferences (Bıyık *et al.*, 2020). The uncertainty quantification of the GP is employed to generate active queries to the user, increasing data efficiency.

Mészáros *et al.* (2022) and Franzese *et al.* (2021b) propose to use GPs for motion learning. The motion is supported by minimizing the epistemic uncertainty of the GP, which is also used to interactively update the database.

6.3 Gaussian Mixture Model

While in GP the hypothesis is that the labels are sampled from an infinite dimensional Gaussian distribution, in a Gaussian Mixture Model (GMM) the hypothesis is that the input and output pairs are samples of a joint distribution defined by a superposition of m Gaussians of dimension $(j + k)$, being j and k the dimension of the input and output features, respectively. The prediction is obtained as a conditioning of the joint distribution on the provided input. Chernova and Veloso (2009) use GMMs for classifying discrete actions and for generating an active and interactive agent, which queries the user in uncertain situations. Moreover, it also allows the aggregation of new labels, which are used for retraining the classifier. In that work, the GMMs are used for quantifying aleatoric uncertainty i.e., the uncertainty in the data. This type of uncertainty is commonly employed to capture the variability of the demonstrations or to spot conflicting labels.

6.4 Support Vector Machine

Support Vector Machine (SVM) is a supervised classification algorithm that maps each data point to an n-dimensional space and identifies the hyperplane that best separates the points belonging to different classes.

In the context of interactive learning, Laskey *et al.* (2016) combine the use of SVM with DAgger for the classification of the states as risky according to the proximity to the decision boundary of the SVM. The use of this strategy shows a reduction of the asked query to the user when performing interactive learning.

6.5 Neural Networks

In general, NNs correspond to a family of function approximators where the approximation is achieved through a composition of multiple operations, known as *layers* (Goodfellow *et al.*, 2016). Currently, NN are known as a tool that can scale well in terms of size, input/output dimensionality, and amount of training data (Goodfellow *et al.*, 2016), which opens opportunities for new technologies and theory. The advances

that NNs have experienced in the past decade created a turning point in different research fields, and IIL has not been an exception.

Despite the opportunities that NNs present, different challenges must be overcome to use them successfully, where requiring large amounts of data, vanishing gradients, and catastrophic forgetting are some of them. Hence, several IIL works focus on unfolding the advantages of NNs while overcoming these different challenges (Warnell *et al.*, 2018; Mandlekar *et al.*, 2020; Pérez-Dattari *et al.*, 2020; Arumugam *et al.*, 2019).

Inspired from the RL literature, several IIL methods employ *replay buffers* throughout the agent's learning process to avoid locally overfitting to the last set of feedback signals and to use the learning data more efficiently (Arumugam *et al.*, 2019; Warnell *et al.*, 2018). However, to use replay buffers successfully, the methods must be able to learn off-policy, which is discussed in depth in Section 7.

Similarly, another strategy that has been borrowed from the RL literature is to add a penalty in the update rule of the policy to avoid large changes in the policy's parameters each time the policy is updated. Mandlekar *et al.* (2020) includes a penalty loss in a learning from intervention framework where two buffers are used, one that stores the interventions and another one that stores the transitions executed by the learner. Every time the policy is updated, data is sampled from both buffers; hence, the learner improves its behavior from the interventions while avoiding drastic changes in its behavior, as it is also trained from data generated by itself. Chisari *et al.* (2022) extended this idea by including evaluative feedback in the learning framework. Therefore, instead of storing every learner's transition into a buffer, only the transitions with positive feedback are stored, avoiding the policy to learn from its own erroneous behavior.

In contrast, Prakash *et al.* (2020) focus on the challenge that unbalanced datasets can present to NNs. DAgger is employed to collect autonomous driving demonstrations online; however, in such complex scenarios, there are situations that have a low probability of occurrence. Consequently, the dataset that the NN uses for learning is largely occupied by frequent and similar situations and will have few instances of uncommon states. Hence, the network is likely to forget these unlikely

situations and not behave properly when they occur. Prakash *et al.* (2020) propose to detect such situations with a measure of epistemic uncertainty using an ensemble of NNs and increase their likelihood of being sampled from the learning dataset.

Finally, State Representation Learning (SRL) is another technique used to avoid overfitting and speed up the learning process. SRL methods, along with the IIL loss, optimize for auxiliary loss functions that are employed to learn state representations from high-dimensional data (Böhmer *et al.*, 2015). For more information regarding these methods, the reader is referred to Section 5.1.

6.6 Movement-Conditioned Models

When dealing with robotics tasks, commonly, the user aims to teach a desired movement to the robot. For these models, the input can be the progress (or phase) of the movement or the current robot state. The output can be the desired position, velocity or acceleration.

6.6.1 Dynamic Movement Primitive

DMPs (Saveriano *et al.*, 2021) generates a movement as the superposition of an attractor model and a non-linear function (known as a *forcing term* $f(s)$). This forcing term enables the generation of complex trajectories while the attractor leads the robot towards the desired goal.

Schroecker *et al.* (2016) investigated the learning of DMPs with Policy Search (PS) and initial demonstration or interactive corrections. In particular, the teacher gives a set of via points at the beginning of the training or the execution of the training can be stopped and the robot moved to the desired position at a particular time. The parameters of the DMPs are updated every time a correction is provided in order to maximize the probability of actually going towards that new via-point. Alternatively, Celemin *et al.* (2019a) used a different strategy for the correction that does not require stopping the robot. In fact, it allows the user to give corrective feedback on the desired state by modeling the motions with DMPs.

6.6.2 Probabilistic Movement Primitive

A movement may be described through a combination of different primitives. For example, motions reaching toward different objects on a table may have similar starting behavior but, depending on where the object is located, their shape may vary. ProMPs are a variant of Movement Primitives (MPs) that enables the capture of the probability distribution of the different demonstrations (Paraschos *et al.*, 2013), as a combination of multiple primitives. For example, multiple demonstrations with different goals would generate trajectories of variable shapes, with increasing variance towards the end. Then, when conditioning the motion on a point near the end of the trajectories, one primitive would be sampled according to this conditioned distribution.

In the context of Active Learning, it is important to adapt the motion not only from the human feedback, but also to minimize the risk of collision during the interaction with the human. Therefore, a modification of the learned ProMP to perform collision avoidance is introduced by Koert *et al.* (2019).

Alternatively to ProMPs, the variability of the demonstrations can be conditioned on the context, e.g., the goal position, the mass of the manipulated object, etc. This approach is proposed by Maeda *et al.* (2017) using GP and by Kulak *et al.* (2021) using GMMs for interactive learning of the forcing term of a DMP.

6.6.3 Kernelized Movement Primitives

A formulation of MPs, known as Kernelized Movement Primitives (KMPs), is introduced by (Huang *et al.*, 2019), where a multi-output formulation is proposed to embed the variability of multiple demonstrations. Although this method is general, it is also tested in an IIL setting. To adapt the robot's trajectory when the environment changes, the use of a force sensor installed at the end-effector of the robot is used to measure corrective forces exerted by the human.

6.7 Discussion

In this section, an overview of different regression and classification methods is introduced with their application in the context of IIL. An important feature of function approximation methods in IIL is the possibility to perform an online update of the function when interactive feedback is provided, and to be able to deal with conflicting data (e.g., ProMP).

Additionally, the use of bayesian methods, like GPs, provides a quantification of the epistemic uncertainty that has been used for actively querying the user to enhance a safer exploration of the robot and interaction with the human. Finally, recent developments of NN allow dealing with high-dimensional sensory inputs while interactively aggregating data from humans. These advances are opening the possibility for IIL algorithms to be applied to new settings where it was not possible before, such as household environments.

7

On/Off Policy Learning

Machine Learning methods intended to solve sequential decision-making problems (like RL or IL) feature different algorithmic properties related to what kind of data is used for learning, when and how it is generated, and how it is used for updating the policy. Depending on how these questions are approached by the method designer, learning processes could be classified as on/off-line learning and on/off-policy learning.

The chronological evolution of the focus on the way the learning data is generated for IL has been relatively opposite with respect to RL. Initially, the main idea of RL was the autonomous learning of a policy by trial and error, while the agent is interacting with the environment, i.e., collecting the data samples while testing the learning policy. However, in recent years, researchers have dedicated efforts to an additional branch for applying the MDPs properties, and RL concepts, for learning from prerecorded data without further agent-environment interaction, as is the case with offline RL (Levine *et al.*, 2020). On the other hand, IL was studied for many years only to find methods that could replicate behaviors contained in static datasets of expert demonstrations, and only later it has been explored the idea of incrementally collecting data from the teacher who observes the learning agent performance.

Due to the different development of these two learning paradigms, general common concepts have been independently introduced. In this section, a discussion intending to unify the definitions of these concepts given in both the RL and IL literature is presented, while trying to keep the RL definitions as the reference.

7.1 Online and Offline Learning

Depending on when the collection of data used for learning is carried out, the learning methods could be classified into offline or online learning. In RL, the offline learning setting is defined as the situation when *"the agent no longer has the ability to interact with the environment and collect additional transitions using the behavior policy. Instead, the learning algorithm is provided with a static dataset of transitions and must learn the best policy it can using this dataset"* (Levine *et al.*, 2020). In contrast, in the online learning setting, the experience the agent gathers for learning increases with new interactions with the environment, allowing it to improve the current policy.

The projection of these definitions into the world of IL matches completely with the classification of interactive and non-interactive methods. Offline learning covers the standard IL methods that sequentially record demonstrations in a static dataset, and later obtain a policy with the recorded data. Online IL methods cover the group of IIL approaches because they feature the ability to collect more data with a dynamic dataset during learning. Since in IL the data collection depends on a teacher, the continuous feedback sampling of online learning involves the teacher in the loop as it has been defined for IIL.

As mentioned in the introduction of this section and considering the introduced definitions, we could say that RL was initially developed for online learning, and only recently its potential for learning offline has been studied, while IL was first formulated offline, and recently extended to the online setting.

In both RL and IL, the agent learns from the obtained feedback, provided by the environment or teacher intervention, respectively. Both learning paradigms aim at a similar objective in the offline case, since both try to obtain a policy that reproduces the behavior recorded in the

data. In other words, offline RL also tries to imitate the demonstrations collected in a dataset; however, it makes use of a reward function that supports the process of defining which decisions in the demonstrated data are more relevant and which ones are less convenient for attaining the task goal.

7.2 On-policy and Off-policy Learning

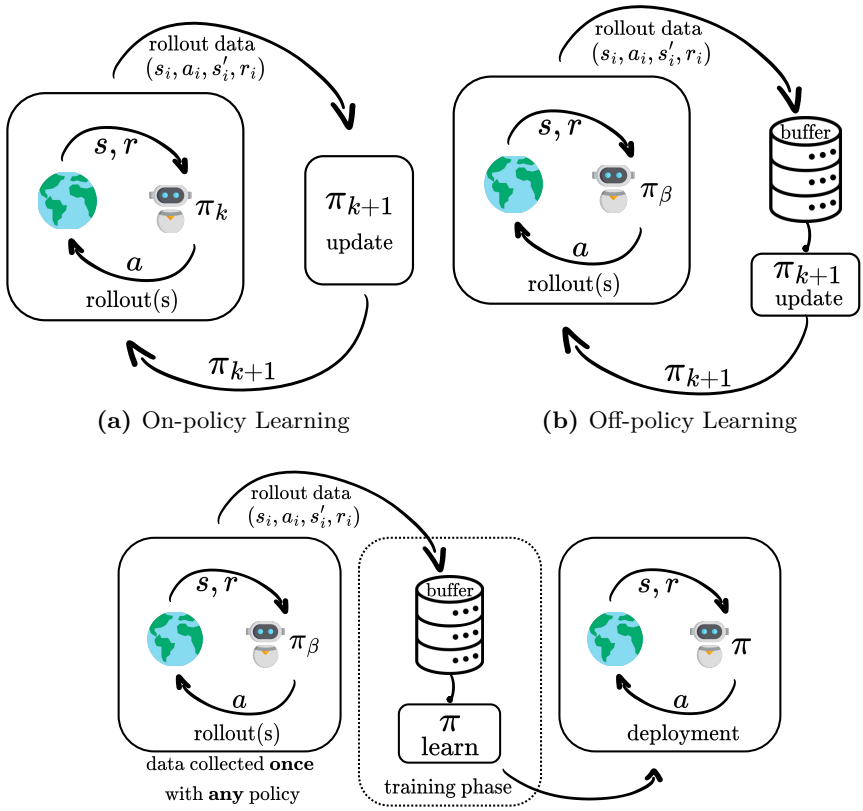
Since, historically, offline learning has been predominately applied in IL methods, two ideas that become evident in online learning scenarios have been mostly ignored in its literature: *on-policy* and *off-policy* learning (see Figure 7.1). These ideas have been well defined and deeply studied by the RL community, and they play a fundamental role in the understanding and design of the learning methods. In this section, we argue that the relevance that on-policy and off-policy learning has in RL also transfers to IIL. However, although some works have used these concepts in the context of IIL (Laskey *et al.*, 2017b; Arumugam *et al.*, 2019; Balakrishna *et al.*, 2020), they are still not clearly defined in this field. Therefore, to analyze the relevance that on/off-policy learning has in IIL, it is necessary to first clearly define it for this case.

Below, we introduce these concepts from the original definitions in the literature of RL, and thereafter they are extended to the IIL case.

7.2.1 On/Off-Policy Learning in Reinforcement Learning

Sutton and Barto (2018) define these concepts stating: *"on-policy methods attempt to evaluate or improve the policy that is used to make decisions, whereas off-policy methods evaluate or improve a policy different from that used to generate the data"*. The policy that is being learned is often referred as *target policy* π^{t}, and the policy used to generate the learning data as *behavior policy* π^{b}. Then, on-policy learning occurs when the learning data comes from trajectories generated by the target policy, i.e., $\pi^{\mathrm{t}} = \pi^{\mathrm{b}}$. In contrast, if $\pi^{\mathrm{t}} \neq \pi^{\mathrm{b}}$, the learning is off-policy. Note that, consequently, offline RL requires off-policy learning.

These concepts can be formally defined from the RL objective and from how it is commonly optimized. From Section 2, Equation (2.1),



(a) On-policy Learning **(b)** Off-policy Learning

(c) Offline Learning

Figure 7.1: Different learning schemes used in RL that are applicable to IL. In on-policy learning, the target policy π_k is the same as the behavior policy. This policy collects the data used in the update that leads to π_{k+1} (a). In off-policy learning the behavior policy π_β is different from the target policy, allowing the use of a replay buffer. However, in practice, π_β results from combining π_k with exploration noise or input from the teacher (b). In offline learning the behavior policy π_β used for obtaining the data is completely different from the policy π obtained in the learning process, which is not considered for the data collection (c). This figure is inspired by Levine *et al.* (2020), with some modifications.

we have that this objective commonly corresponds to the maximization of the discounted expected return

$$\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}_{\tau \sim p_\pi(\tau)} \left[G(\tau) \right], \tag{7.1}$$

where $G(\tau) = \sum_{t=0}^{T} \gamma^t \mathcal{R}(s_t, a_t)$ corresponds to the return.

In practice, since we do not have analytical access to this expectation, to find the policy π^* that maximizes the presented objective, it is necessary to empirically collect information from, ideally, every possible trajectory τ (or transition $(s_t, a_t, s_{t+1}, r_{t+1})$ given the Markov assumption) and shift the behavior of π^{t} towards the trajectory distribution that maximizes Equation (7.1). However, in most realistic scenarios, it is not possible to sample the complete state-action space; hence, a policy is commonly chosen to sample this space as diversely and exhaustively as possible while keeping the problem tractable. This policy is π^{b}. Then, at every update iteration, the data collected by π^{b} is employed to estimate the current expected return of the trajectory distribution induced by π^{t}, and π^{t} is modified such that this expectation increases. However, if data is generated by sampling trajectories induced by π^{b}, *how is the expected return computed with respect to* π^t*?* There are two options, 1) on-policy learning, i.e, directly improve π^{b} at every iteration ($\pi^{\mathrm{b}} = \pi^{\mathrm{t}}$), 2) off-policy learning, i.e., $\pi^{\mathrm{b}} \neq \pi^{\mathrm{t}}$ and employ a strategy to compute the expected return of π^{t} from trajectories collected by π^{b}. Consequently, at every learning iteration, the estimated objective of an on-policy learning method corresponds to

$$\textbf{on-policy: } \hat{\mathbb{E}}_{\tau \sim p_{\pi^{\mathrm{b}}}(\tau)} \left[G(\tau) \right], \tag{7.2}$$

where $\hat{\mathbb{E}}_{\tau \sim p_\pi(\tau)}$ corresponds to the expectation estimated by sampling data from the environment following a policy π.

In contrast, the estimated objective of off-policy methods, even though the data comes from π^{b}, corresponds to

$$\textbf{off-policy: } \hat{\mathbb{E}}_{\tau \sim p_{\pi^{\mathrm{t}}}(\tau)} \left[G(\tau) \right]. \tag{7.3}$$

Note that off-policy methods are defined as those that are able to learn off-policy, which indicates that it is also possible to learn on-policy with these methods, as they are capable of learning from data generated

by any policy, which includes the target policy (Sutton and Barto, 2018).

The RL literature provides a vast family of on-policy and off-policy learning methods, to study how the concepts of on/off-policy are applied in practice we can analyze some examples.

SARSA and Q-Learning

To illustrate the difference between on-policy and off-policy methods, let us study SARSA (Rummery and Niranjan, 1994) and Q-Learning (Watkins, 1989), two seminal RL methods. SARSA is on-policy and Q-Learning is off-policy. These methods employ Temporal-Difference (TD) learning to compute estimates of the expected return and solve Equation (7.1). TD combines ideas from Monte Carlo (MC) methods and Dynamic Programming (DP), i.e, trajectories are empirically sampled from the environment, but the final outcome is estimated based on current models of the environment (which is known as *bootstrapping*), instead of only using the sampled data. SARSA and Q-learning employ TD learning to estimate the action-value function $Q(s_t, a_t)$, which estimates the expected return of a policy given its current state and selected action and derive a policy from it. Hence, the environment can be sampled following π^{b} and bootstrapped at every time step to get the following sample/estimation of the return:

$$G(\tau)_t = \underbrace{r_{t+1}}_{\text{sample}} + \underbrace{\gamma Q(s_{t+1}, a_{t+1})}_{\text{estimation}}. \tag{7.4}$$

Then, Q can be updated by computing the error of this TD estimate with respect to the current estimation of Q for a given time step, which is known as the update rule of SARSA:

$$\textbf{SARSA: } Q^{\text{new}}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \underbrace{[\underbrace{r_{t+1} + \gamma Q(s_{t+1}, a_{t+1})}_{\text{TD target}} - Q(s_t, a_t)]}_{\text{TD error}}, \tag{7.5}$$

where α is the learning rate of the update. Note that the only variable that indicates that Equation (7.5) is following π^{b} is a_{t+1}, as the other variables are a consequence of the action taken by π^{b} one time step

before, which can be ignored at $t+1$ given the Markov assumption. Hence, it is possible to remove the dependence of the TD-target from π^{b} if instead of using the action a_{t+1} sampled from π^{b} in this estimate, a different one is chosen. This idea can be followed to create an off-policy variation of SARSA, known as Q-learning.

Q-learning defines its target policy as the optimal policy according to the current estimation of Q, i.e., $\pi^t(s_t) = \arg\max_a Q(s_t, a)$. Then, Equation (7.5) can be modified by replacing the term $Q(s_{t+1}, a_{t+1})$ with the Q value of π^{t}, making the return estimation to be according to π^{t} instead of π^{b}, i.e,

Q-learning:

$$Q^{\mathrm{new}}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)], \quad (7.6)$$

where the modified value with respect to Equation (7.5) is highlighted in red. Note that in the special case where the behavior policy is greedy with respect to the current estimate of Q (e.g., when using an ϵ-greedy exploration strategy and ϵ tends to zero), the SARSA and Q-Learning update rules are equivalent (Rummery and Niranjan, 1994) because the target and behavior policies are the same, i.e., Q-Learning becomes on-policy.

Importance Sampling

Another well-known approach for designing off-policy learning methods is *importance sampling*. Importance sampling allows methods that in nature are on-policy, such as SARSA, to become off-policy by weighting the TD errors with the importance sampling ratio (Sutton and Barto, 2018; Mahmood, 2017). The importance sampling ratio is employed to estimate the update of the Q function of the target policy from data generated by a different policy, e.g., the behavior policy. Closely related to the methods studied above, the method Temporal-Difference per Decision Importance Sampling (TD-DIS)[1] (Precup, 2000) can be

[1]The acronym TD-DIS is introduced in this work for simplicity, given that no acronym is proposed in (Precup, 2000).

analyzed in this case. TD-DIS method can be seen as an off-policy extension of SARSA by means of importance sampling (Rubinstein and Kroese, 2016; Hammersley and Handscomb, 1964). The update rule of TD-DIS is

TD-DIS:

$$Q^{\text{new}}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\rho_t[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)], \quad (7.7)$$

where

$$\rho_t = \frac{\pi^t(a_t|s_t)}{\pi^b(a_t|s_t)} \quad (7.8)$$

is the per-step importance sampling ratio between the target policy and the behavior policy. The similarities between SARSA and TD-DIS are evident; ρ_t is the only variable that differentiates both methods and allows the update rule of TD-DIS to be employed with data collected by the target policy. In the special case where this method is used on-policy, the behavior and target policies become the same (i.e., $\rho_t = 1$) and TD-DIS becomes equivalent to SARSA.

7.3 On-Policy/Off-Policy Learning in Imitation Learning

According to Osa *et al.* (2018), the terms on-policy and off-policy, in the IL literature, are mentioned for the first time in Laskey *et al.* (2017b). The authors use the terms on-policy and off-policy according to which policy is used to sample data from the environment. If the current agent's policy is used to sample data, then the method is on-policy; if the teacher's policy is used to sample data, then the method is off-policy. Although this definition may seem equivalent to the one in RL, there is a difference: in RL, these definitions are about the data that is used in the evaluation or improvement of the *current* agent's policy.

This difference is important because online RL and IIL are iterative learning processes (i.e., the agent's policy is evolving over time while it interacts with the environment), which means that the distribution of the data generated by an older version of the agent's policy is not the same as the one generated by the current agent's policy. The on/off-policy definitions provided in Laskey *et al.* (2017b) allow on-policy methods to use data generated by older versions of the agent's policy

(i.e., other policies) when improving its behavior, which is not consistent with the RL definition.

As an example, DAgger (Ross *et al.*, 2011) has commonly been defined as being on-policy and Behavioral Cloning as off-policy (Osa *et al.*, 2018; Laskey *et al.*, 2017b; Balakrishna *et al.*, 2020). Nevertheless, in DAgger, data is constantly being aggregated in a dataset that is used to update the agent's policy iteratively. Consequently, data generated with a different policy than the target policy is used in the update rule, and, therefore, from an RL perspective, it would be an off-policy method. From this point of view, DAgger and Behavioral cloning are in the same category.

Instead, we argue that the ideas of on/off-policy learning can be transferred differently to IIL. In this section, we focus the analysis on the per-step feedback case as described in Section 2.2.2, as it is the case that most resembles RL.

7.3.1 From Reinforcement Learning to Interactive Imitation Learning

From the definition of on/off-policy learning in RL provided in Section 7.2.1, we can recall that off-policy learning occurs when data collected with one policy (i.e., behavior policy) is employed to update a different one (i.e., target policy). Consequently, off-policy learning allows updating a policy from data that has no dependence on it.

This same idea can be employed to study on/off-policy methods in IIL, i.e., if the data used in the update rule of the learner's policy follows a different policy, the learning method is off-policy; otherwise, it is on-policy. The only difference is that, in this case, the learner collects the feedback signal when interacting with the environment, instead of the reward signal like in RL. As mentioned in Section 2.2.1, the feedback signal can be understood as a generalization of the reward.

To observe this more clearly, we can analyze methods from the two paradigms that lead to IIL methods (see Section 2.2.4): 1) Value Maximization and 2) Divergence Minimization.

7.3.2 Value Maximization Methods

Since these methods derive from the RL literature, they optimize the RL objective, and, therefore, the definitions provided in Section 7.2.1 can be directly used to define them as being on-policy or off-policy. Let us study two of these methods: COACHe (MacGlashan *et al.*, 2017) and TAMER (Knox and Stone, 2008).

COACHe COACHe is derived employing the *policy gradient theorem* of RL (Sutton and Barto, 2018). This theorem allows to directly improve the parameters θ of a policy π by computing the gradient of its value function and applying stochastic gradient ascent. Consequently, COACHe applies the following update rule to its policy:

$$\theta^{\text{new}} \leftarrow \theta + \alpha \nabla_\theta \pi(s_t, a_t) \frac{h_{t+1}}{\pi(s_t, a_t)}, \tag{7.9}$$

where α is the learning rate and h_{t+1} the human feedback. Here, h_{t+1} can be interpreted as replacing the *advantage function* used in this type of policy gradient methods, which describes how much better or worse an action would perform compared to the agent's action when following the agent's policy. Consequently, to improve the agent's policy with this method, it is necessary to learn **on-policy**; otherwise, h_{t+1} would indicate the advantage of an action with respect to a policy different from the agent's policy, making its update incorrect.

TAMER TAMER can be interpreted as a method that maximizes the Q function for deriving a policy but assumes that the policy behaves *myopically* (i.e., $\gamma = 0$). Therefore, we can observe that if we assume a myopic behavior, Eqs. (7.5) and (7.6) reduce to the same solution, which corresponds to the update rule employed by TAMER

$$Q^{\text{new}}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[h_{t+1} - Q(s_t, a_t)], \tag{7.10}$$

where the reward r_{t+1} is replaced by the human feedback h_{t+1}. Consequently, TAMER interprets the feedback signal as a Q-value, which does not depend on the agent's policy (Zhang *et al.*, 2019b). Moreover,

it does not depend on *any* policy, but only on immediate actions. Consequently, with TAMER, it is possible to update the target policy with data collected by any policy, making it an **off-policy** learning method. Note that although it is likely that the teacher will provide feedback as a function of the learner's policy (Knox and Stone, 2009; MacGlashan *et al.*, 2017). For instance, TAMER is proposed considering assumptions such as *"The trainer can evaluate an action or short sequence of actions, considering the long-term effects of each"* (effects with respect to the policy) and *"a human trainer's reinforcement function, is a moving target. Intuitively, it seems likely that a human trainer will raise his or her standards as the agent's policy improves"*. However, from an algorithmic perspective, given the proposed implementation, the feedback is policy-independent.

7.3.3 Divergence Minimization Methods

Let us recall Equation (2.7), which summarizes IIL methods based on Divergence Minimization and rewrite it in its MLE form (Ke *et al.*, 2020), instead of the Kullback–Leibler (KL) divergence between two policies; then, in each training iteration we solve

$$\max_{\pi \in \Pi} \mathbb{E}_{s \sim p_\pi(s), a \sim \pi^h} \left[\ln(\pi(a|s)) \right]. \tag{7.11}$$

This form of the equation is useful for analyzing the on/off-policy nature of these methods because it explicitly shows the state and action distributions. Here, we analyze two IIL methods derived from this equation: DAgger (Ross *et al.*, 2011) and COACHc (Celemin and Ruiz-del-Solar, 2019).

DAgger In DAgger, Equation (7.11) is minimized by setting the feedback signal to $h_t = a_t$. Hence, h_t directly corresponds to a label of the optimal action (according to the teacher) for a given state. This method assumes that for every sampled state from which this equation is minimized, the label h_t does not depend on the current agent's policy, as it only follows π^h. Therefore, it is possible to update the agent's policy from data generated by any policy, which makes DAgger an **off-policy** learning method.

Nevertheless, there is one important remark to make. Given that the state distribution of the samples used to update Equation (7.11) depends on the behavior policy, a different behavior policy will, inevitably, yield different solutions. However, this is also the case for RL methods, so this definition is still consistent with them.

COACHc In COACHc, the assumption is that the feedback signal corresponds to an error signal that indicates the direction in which the current agent's policy should be modified to improve its performance (feedback is only meaningful for improving the current behavior policy, and not future versions of it). Therefore, in this case, for every training iteration, an action label is generated as a function of the learner's policy with the form $a = \pi^l(s) + e \cdot h_t$, where e is a hyperparameter defined as the error magnitude. Consequently, Equation (7.11) gets modified, since the actions do not distribute as π^h anymore, but rather as a different distribution that depends on π^l. Therefore, it is only possible to update COACHc with samples collected by the agent's policy, making this method an **on-policy** learning method.

7.4 Discussion

The concepts discussed in this section are as important in IIL as in RL because they are agnostic of the feedback source used for policy improvement (teacher or environment). Instead, they are related to the way the learning experience is obtained and used in the policy updates.

The replay of recorded experience and the way it is implemented is one of the main features that come into the discussion of On/Off-policy learning. But unlike RL, wherein the reward function (that could be deterministic or stochastic) is (time or policy) invariant, IIL methods could have in some cases feedback of the teacher that depends on the performance of the policy. Therefore, depending on the assumption about the teacher's feedback within a learning method, it is relevant to evaluate what kind of learning is the most convenient for the method implementation, such that it leverages that assumption.

Since in online learning the experience is incrementally collected, there are additional challenges when fitting function approximators with this kind of data. The sequential nature of these problems makes

the data have spatio-temporal correlations, therefore not following the IID assumption of most ML approaches. Additionally, when training NNs from a static dataset, the iterative process of updating the model normally reduces the error for most training data as long as there are sufficient iterations. That is because some data points require more update steps following the cost function gradient than others. However, when data samples are obtained incrementally while also learning, it is difficult to control the model to avoid either overfitting or underfitting the data. In both cases, there is the additional issue of the model being changed for other input-output mappings different from the ones used in the update, which counts as losing the already acquired knowledge, and is known as "catastrophic forgetting".

Experience replay is a technique introduced for breaking those correlations in the collected data during the policy update. It also helps to have a good balance for not overfitting to the most recent training data, while keeping the old experience in the *memory* of the model, i.e., it helps to deal with the three problems previously mentioned.

For instance, in methods wherein the teacher provides evaluative feedback at any time step, there could be two different cases:

1. When the human feedback is assumed to replace the MDP reward and used within an RL implementation, the feedback is assumed to be consistent in all the state-action space, such that the RL learning properties hold. In this case, the old feedback samples are never conflicting with the new ones, therefore, old feedback signals are always usable, and the choice of on/off-policy learning is left to the RL implementation, being both valid.

2. When it is assumed humans consider past and future in their evaluative feedback signals, and it is used as something equivalent to value function (e.g., TAMER), i.e., the feedback depends on the policy. Consequently, feedback given over state-action pairs of old policies could be contradictory with respect to the one obtained with the current policy. This assumption requires giving priority to the feedback given to the execution of the current policy, hence on-policy learning would be more appropriate.

Since the discussions of On/Off-policy learning are relatively new and not consolidated in IL, this dimension of the algorithmic features space has been neglected in some implementations of IIL methods. Some algorithms have considered a learning strategy that does not align with the assumptions of the required human feedback. It is not simple to implement methods whose algorithmic features match the feedback assumptions because the limitations created by the aforementioned problems (non-IID, over/under-fitting, catastrophic forgetting) condition the learning strategies.

The most common case of having inconsistent implementations is when the feedback is policy-dependent, but experience replay is required for stable learning, i.e., on-policy learning deals better with the assumed policy-dependent feedback, but the need for experience replay makes it necessary to learn from off-policy data. As mentioned before, importance sampling helps for decreasing the priority of data obtained with different policies Degris *et al.*, 2012, which is convenient for learning from off-policy data with methods whose assumptions align with the on-policy learning conditions. The Convergent Actor-Critic by Humans (COACHe) algorithm Arumugam *et al.*, 2019 is a good example of IIL with a policy-dependent feedback assumption (naturally on-policy), which benefits of off-policy learning for stability, but using importance sampling to prioritize the data in the updates according to the distribution of the current policy.

8

Reinforcement Learning with Human-in-the-Loop

As explained in the Theoretical Background section, RL represents a learning framework where an *autonomous agent* learns the desired behavior by interacting (in a form of *trial-and-error*) with the *environment* that provides the *reward* (as a feedback signal). Defined as such, RL offers a general framework and, in general, all other entities except the autonomous agent (e.g., other agents/humans) can be considered as a part of the environment. However, considering *humans* as a distinct entity from the environment enables the design of algorithms that take advantage of this setting. There are some RL problems that also consider interactions with humans that act in the environment (e.g., human collaboration). In contrast to these, in this section, we focus only on those problems where a human is influencing the *learning loop* by providing feedback to the agent (e.g. has the role of an observer that provides feedback), and not just acting in the same environment as the agent. In these cases, we refer to the human as a teacher, as explained in Section 2.

Looking from a RL perspective, if the teacher is influencing the learning loop and it is the only one providing feedback to the agent (i.e., demonstrations, corrections, etc.), the problem reduces to the

IIL problems. If an autonomous agent receives a feedback signal from both the environment and the teacher, it is learning with a *RL with Human-in-the-Loop* method. We define as RL-HiL all RL algorithms where the agent learns a desired behavior and a teacher is influencing one or more components of the reinforcement learning loop (e.g. reward, policy, exploration, etc.).

8.1 Other Related Approaches

As it was already discussed in Section 1, several RL approaches rely on human demonstrations. For instance, the approaches that pre-train a policy to warm-start the learning process, storing pre-recorded demonstrations for either off-policy RL or offline RL, or inferring rewards from demonstrations like in IRL. However, we do not consider these as RL-HiL as the human input is not part of the learning loop. Several approaches rely on human demonstrations to initialize the policy and subsequently use RL to further improve it (e.g., Policy Learning by Weighting Exploration with the Returns (PoWER) (Kober and Peters, 2008), Locally Optimal search after K-step Imitation (LOKI) (Cheng *et al.*, 2018)). Pre-training the policy in a supervised manner from demonstration data before interacting with the real task is especially important for alphaGo (Silver *et al.*, 2016), to have a reasonably good policy for starting self-play. Deep Q-learning from demonstrations (DQfD) (Hester *et al.*, 2018) also pre-train the network with expert demonstrations and keeps them in a replay buffer, along with the data obtained from its own experience.

8.2 Historical Perspective

Historically, RL-HiL evolved with RL research from the beginning. If we look for the origins of the idea of using teacher advice, we can find that they go even to the roots of founding the AI field, with John McCarthy's *Advice take* (McCarthy, 1958) that was proposed to improve behavior by taking pieces of advice. Some works already in the 90s try to combine Reinforcement Learning approaches with human advice (Utgoff, 1991; Whitehead, 1991; Clouse and Utgoff, 1992; Lin,

1992; Maclin and Shavlik, 1994). Utgoff (1991) is mainly inspired by advancements in the Checkers game-playing agents. The initial game-playing agent was based on *learning from experience* (Samuel, 1959) and extended later with utilizing expert choice, typically called a *book move* (Samuel, 1967). The main idea by Utgoff (1991) is that these are two fundamentally different kinds of training information useful for learning. Additionally, they argued that expert choice does not have to be recorded in a *book* in advance. By watching an expert in action, or asking an expert what to do in a particular situation, an agent would be able to learn from an expert whenever it is available. The agent is able to query an external agent about the correct action to take when the confidence in its own decisions is low. Their agent is used to control a search algorithm by considering as an action which node to explore next, like in the checkers engine (Samuel, 1959). Clouse and Utgoff (1992) extend the idea beyond a search algorithm to the cart-pole task and provide an online mechanism that allows an external agent to guide it while it performs the task.

On the other hand, Whitehead (1991), inspired by the idea that RL can be viewed as an online search where an agent explores unknown environments instead of a simulated model, extend blind exploration (like in blind search e.g. Dijkstra's algorithm) with two cooperative mechanisms: Learning with an External Critic (LEC) and Learning By Watching (LBW). In the work by Maclin and Shavlik (1994), an *advice-giver* watches the learner and occasionally makes suggestions to help it, expressed as instructions in a simple programming language (e.g., rules to avoid an enemy). These pieces of advice are integrated and refined by RL.

As one of the most recognized examples of RL-HiL approaches, Isbell and Shelton (2001) present Cobot, an application of RL for LambdaMOO, a complex, open-ended, multi-user chat environment that can take proactive actions and adapt the behavior from multiple sources of human reward. After 5 months of crowd-sourced training, and 3171 reward and punishment events from 254 different users, Cobot learns nontrivial preferences for a number of users.

8.3 Reinforcement Learning with Human-in-the-Loop Approaches

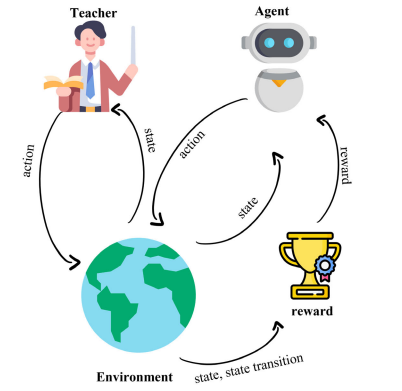


Figure 8.1:
Human-guided exploration

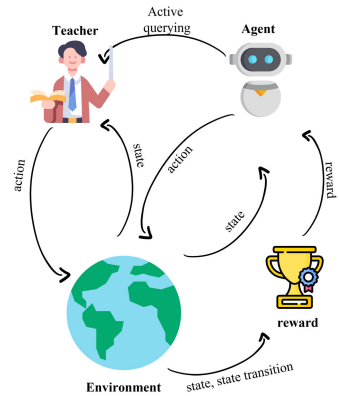


Figure 8.2:
Human intervention for Safe-RL

A few surveys on the topic of RL-HiL have been recently published, providing a deeper insight into the field (Zhang *et al.*, 2019b; Arzate Cruz and Igarashi, 2020; Lin *et al.*, 2020; Najar and Chetouani, 2021). Although there is no consensus, one way of clustering these approaches is based on the way human feedback is integrated into the learning loop (e.g. as a reward, as a policy correction, as a guiding policy for exploration, etc.). Based on that, RL-HiL approaches can be clustered into four main categories:

- Human-guided exploration, see Figure 8.1
- Human intervention for Safe-RL, see Figure 8.2
- Reward shaping with Human feedback, see Figure 8.3
- Policy shaping, see Figure 8.4

which we briefly summarize here.

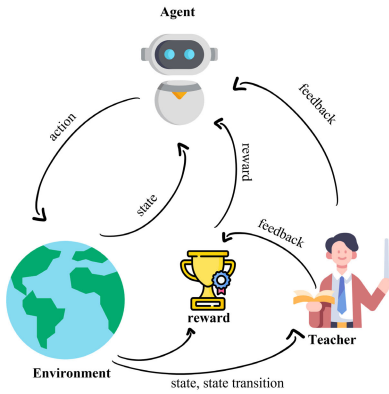


Figure 8.3:
Reward Shaping

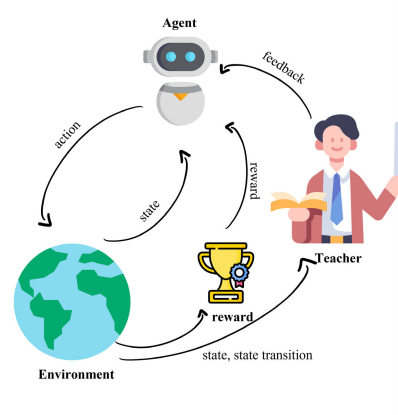


Figure 8.4:
Policy Shaping

8.3.1 Human-guided Exploration

As known from many RL applications, training RL agents requires extensive trial-and-error experience. A necessary component for improving the policy is *exploration*. To accumulate high returns, the agent needs to follow actions that were leading to high returns in the past. On the other hand, the agent sometimes needs to try-out actions that it did not try before. This, ideally, allows it to find strategies that obtain higher returns and, therefore, improve the policy. There are several environments that are particularly hard for learning from the perspective of exploration, mainly *sparse-reward* environments and environments containing *fatal failures*.

Sparse reward functions normally return a constant signal for most of the states, while they provide a different value in a few states, normally for indicating success and/or failure. One example of such an environment is a puzzle, where the agent receives a reward only once the puzzle is successfully solved. Obtaining useful information from random exploration in sparse reward settings makes learning challenging. Therefore, utilizing human feedback for guiding the agent during the exploration phase has been proposed to avoid this problem.

In Suay and Chernova (2011), a human provides rewards but also anticipatory guidance on the selection of future actions. The users can guide during a short period before actions are selected by the robot, such that they limit the set of actions that can be explored in a specific state. This can be interpreted as directing the robot's attention towards the target object or area. Subramanian *et al.* (2016) present a model-free policy-based approach called Exploration from Demonstration (EfD). The user feedback is used to shape an exploratory policy and bias a RL agent's exploration to cover the search space effectively. The algorithm can query for demonstrations from the user by highlighting the states in a Graphical User Interface (GUI). The user can help the learner by providing demonstrations (reaching the states) using the GUI. The user can alternatively choose to ignore the query or stop interacting with the algorithm. These demonstrations are used to learn an exploration policy to reach desired states.

In some applications (e.g., letter writing), it was shown to be useful that users can select some desired regions to direct exploration of the learning algorithm. Schroecker *et al.* (2016) enable to interactively add *via-points* that restrict the search space to trajectories that will be close to the defined via-points at the specific time. This in turn significantly reduces the number of samples necessary to learn a good policy. Besides approaches where the teacher is directly guiding an exploration policy, there are also approaches where it is done in an indirect way, such as using gaze or natural language. Saran *et al.* (2021) propose an approach that utilizes indirect information gained from a human observer by detecting the human gaze, while the agent is interacting with the environment. The algorithm guides the agent to explore the regions that are targeted by the human gaze. Human corrections can also be considered as the exploration disturbances of a stochastic policy in Policy Search approaches (Celemin *et al.*, 2019a; Celemin *et al.*, 2019b), or as guidance in which the RL agent memorizes the advice on where to explore (Scholten *et al.*, 2019).

8.3.2 Human Intervention for Safe-RL

There are environments in which the safety of the system is not guaranteed for all possible transitions in the state space. Learning policies in such scenarios with a high risk of failure poses a difficult challenge, especially for executing exploratory actions. Such problems are the focus of the research of Safe Reinforcement Learning (Safe-RL). As seen in a Safe-RL survey (Garcıa and Fernández, 2015), one approach to Safe-RL is to use external knowledge in a form of teacher advice, e.g., the learner agent asks for advice when the confidence level is low (Utgoff, 1991). Safety can be guaranteed using shields (Alshiekh *et al.*, 2018), which are models that prohibit actions leading to catastrophic failures. However, shields are usually conservative and might limit exploration. Designing a shield that is not conservative but still safe is rather challenging. Some approaches utilize *teacher-oversight* to interactively learn a model of a safety shield (Marta *et al.*, 2021). A similar approach (although not based on standard RL) is presented by Kahn *et al.* (2021), where a model of human intervention probability is learned from human interventions. It is later used in a model predictive control fashion to find actions that have a lower probability of human intervention.

8.3.3 Reward Shaping with Human Feedback

Besides the exploration process, human teachers can provide insights into the long-term benefit of an action. This can be done in the form of a reward signal in addition to the one coming from the environment. The idea of *reward shaping* consists of designing a reward function that not only defines the main objective but also provides useful guidance for effective exploration (Ng *et al.*, 1999). However, designing a reward function in advance that balances the goal definition and the guidance task is often very difficult, and the agent might get stuck in a local maximum, e.g., exploiting the local reward without reaching the goal, also known as *reward hacking* (Amodei *et al.*, 2016). The human-provided reward can be also considered as a type of reward shaping. It can be used to guide the agent to learn the task faster. As it is provided interactively, human feedback can be even used to correct the negative effects of

reward hacking. If the user notices the agent is exploiting the reward used for guidance without progressing on task execution, it can locally correct it by adding a negative reward.

One of the first approaches of this family is Interactive RL, as presented by Thomaz *et al.* (2005). A human teacher can, in real-time, provide a reward signal in addition to the reward provided by the environment. TAMER (Knox and Stone, 2008), is also used in an extension with RL, where TAMER is used first, and then the policy is fine-tuned with RL (Knox and Stone, 2010; Knox and Stone, 2012). Similarly, Xiao *et al.* (2020) present Feedback-based REward SHaping (FRESH), an approach where a human is presented with trajectories from a replay buffer and then provides feedback on actions (good or bad) or even on states. A NN is then trained to generalize this feedback to unseen states and actions, and the feedback of the NN is converted to shaping a reward that augments the reward provided by the environment. Recently, Arakawa *et al.* (2018) introduce an algorithm called DQN-TAMER as a combination of TAMER and DQN. The method learns two Q functions in parallel, one from the environment reward and one from the user feedback. Then, the action is computed as the action that maximizes the combination of both functions. This approach is not directly Reward Shaping but rather Value Shaping.

8.3.4 Policy Shaping

There are also works combining RL methods and teachers in the learning loop for directly obtaining an explicit policy representation. This has been studied especially from the perspective of using absolute corrective feedback as in DAgger algorithms. Methods like Aggregate Values To Imitate (AggreVaTe) (Ross and Bagnell, 2014) extend the incremental collection of demonstrations with the use of cost (reward) functions provided by the environment, which are considered for improving the policy on top of the information provided by the teacher. Chang *et al.* (2015), propose a similar learning scheme, although their paper is focused on the problem of learning from a *poorly performing reference policy* (teacher), showing that the feedback of the environment helps to improve the knowledge obtained from the teacher. Later on, Sun *et al.*

(2017) extended AggreVaTe for using it with differentiable policies, such that it is possible to use powerful expressive models (e.g., deep NNs).

This line of research intends to leverage the benefits of both IL and RL worlds, similarly to the domains of other sections. In order to be able to obtain policies outperforming the teacher, it can be more effective to optimize the environment objective function based on the demonstrated trajectories, instead of optimizing the discrepancy between the learner and the teacher. Therefore, such a strategy can relax the requirement of having expert or near-optimal demonstrators in the learning loop.

8.4 Discussion

This section provides a concise, non-exhaustive overview of the RL-HiL field from the RL perspective. Human-in-the-loop approaches have a long history in the field of RL, and in many applications, they are critical for efficient and safe learning. From IIL perspective, they offer an approach to overcome suboptimal human feedback. The approaches can be analyzed based on the way the human input is integrated into the learning loop, making the distinction between approaches improving exploration (efficiency and safety), approaches modifying the reward and approaches directly improving the policy.

9

Interfaces

In an interactive learning setup, the human teacher and the robot are frequently exchanging information. It is natural for humans to communicate in many different forms, making discussions about *how* and *what* to communicate with the robot to be often overlooked. However, from the robot's perspective, the type, the format, and the content of the received information can greatly impact its learning process. Furthermore, choosing an adequate way for the robot to communicate with the human can improve the latter's understanding of the system, which will in turn improve the quality of information the teacher provides to the robot.

While Section 3 discusses the types of feedback that the teacher can provide, this section focuses on the channel used for exchanging this information. According to Dudley and Kristensson (2018), an interface is the bridge in charge of the bidirectional feedback between the user and the system, as illustrated by Figure 9.1. Here, we define interfaces as the physical channel used to capture/provide data together with required software that processes the raw data into the desirable information type.

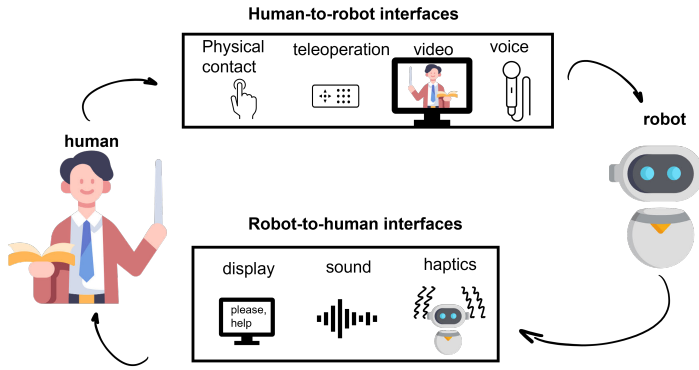


Figure 9.1: Human-to-robot interfaces and robot-to-human interfaces commonly employed in IIL.

First, we summarise the different interfaces used by humans to provide feedback, highlighting which type of information is more appropriate to provide using each interface, and the respective learning methods that can efficiently explore that information. Second, we review interfaces used by the robot to provide information to the human, highlighting which information is provided back to the human and how it is used in the learning loop.

9.1 Human-to-Robot Interfaces

In IIL, most methods are based on passive learners, which means the interactions are only sent from the teachers towards the agent, without having active responses or queries from the robot. The users transfer their knowledge to the learner through the feedback signals using different kinds of interfaces, as discussed in this Section.

9.1.1 Physical Contact with the Robot Embodiment

Being able to teach the robot by physically moving it is a promising method to teach generic tasks since it does not require any extra interface with the user (Ben Amor *et al.*, 2009). This approach is generally called kinesthetic teaching. Kinesthetic teaching is especially motivated by its

simplicity, which is a key feature to enable the employment of robots by non-expert users in everyday tasks.

Teaching trajectories is arguably the most common use of kinesthetic input, which allows the user to provide demonstrations in a natural manner (Wrede *et al.*, 2013). For example, kinesthetic information can be used to learn user preferences (Jain *et al.*, 2013; Canal *et al.*, 2016; Bajcsy *et al.*, 2017; Bajcsy *et al.*, 2018; Losey *et al.*, 2022) such as distance to objects for safer manipulation. Learning Interactively to Resolve Ambiguity (LIRA) (Franzese *et al.*, 2021a) shows how it is possible to solve ambiguities from kinesthetic demonstrations; when the learning agent recognizes an ambiguous situation, it requests the user to move the end-effector towards the correct action, allowing it to learn the correct one. FERL (Bobu *et al.*, 2022) presents a method to learn non-linear features from kinesthetic demonstrations of partial-trajectories. Such features allow for improving learning efficiency and generalization capabilities.

Kinesthetic teaching has been successfully applied to robotic arms; however, such an interface might not be feasible or safe in other scenarios. For example, the robot's size has to be within the human manipulability spectrum, making it not applicable for nano- or industrial-sized robots. Also, high-speed robots render kinesthetic teaching infeasible and/or unsafe, especially on tasks requiring high-speed controlling (e.g., balancing tasks). Due to such limitations, in some learning settings, other interfaces can be more adequate for teaching robots interactively.

9.1.2 Physical Contact with External Device

Keys are present in most computational units (e.g., keyboards, joysticks, remote controllers, touch pads) and are arguably the most standard way for users to input information into systems. Key-based interfaces are convenient since they are readily available in most systems and can be mapped to provide different types of feedback to robotic systems. Keys can be used to provide full demonstrations as shown by Palan *et al.*, 2019.

Even though many IIL approaches use keys for their convenience to create proof-of-concept tests using simulations (Thomaz and Breazeal,

2007a; Argall *et al.*, 2011a; Akrour *et al.*, 2014; Spencer *et al.*, 2020; Zhang *et al.*, 2020; Myers *et al.*, 2022), games (Knox and Stone, 2009; Subramanian *et al.*, 2016; Christiano *et al.*, 2017; Warnell *et al.*, 2018; Xiao *et al.*, 2020; Jauhri *et al.*, 2021), or real robots (Argall *et al.*, 2008; Suay and Chernova, 2011; Knox *et al.*, 2013; Kelly *et al.*, 2019; Hoque *et al.*, 2021), there are three points one must consider about key-based inputs.

First, it is undesirable to have an interface with many keys available in IIL, complicating the usage of the system. As such, keys are better used on systems with limited user input requirements, such as learning from human reinforcements (see Section 3.1.1) or from human preferences (see Section 3.1.2).

Second, the meaning of each key can be programmed at a software level, namely *mapping the keys*. Such mapping can be task-specific, which is common in cases the keys are used to provide feedback in the state/action domain (see Section 3.2); for example, Ross *et al.* (2011) associates keys to actions in a video game (left, right, jump, speed). However, such mappings can also be independent of the task, which is the case for learning from human reinforcements (see Section 3.1.1); for example, TAMER (Knox and Stone, 2008) expects binary feedback that indicated the optimality of the policy, making it task agnostic.

Furthermore, aiming to improve the inherently limited amount of information that keys can carry, Loftin *et al.* (2016) propose to use a statistical user-feedback model, which allows extrapolating cases in which the user did not provide feedback.

An option to encode more information in a single input is to use analog sticks (joysticks), which allow mapping multiple values depending on how far the stick is displaced by the user. They are commonly used in driving scenarios due to their precision for controlling continuous values (Corrigan *et al.*, 2016). For instance, an analog stick is used by Ross *et al.* (2011) to provide demonstrations of the correct steering of a kart in a racing game to train with the DAgger framework. A digital sliding bar is used by Wilde *et al.* (2022) in order to provide an analog value for user preferences in a Learning from Preferences (LfP) setup.

Despite the above considerations, 6 DoF interfaces (e.g., space mouses, cellphones) allow the user to directly control the robot's end-

effector position and orientation directly (Luo *et al.*, 2021), and to give continuous signals (as analog sticks). Such interfaces allow interactive teaching in scenarios where the human cannot provide kinesthetic demonstrations. For example, Celemin *et al.*, 2019a propose to use keys to train a robotic arm to perform the *ball-in-a-cup* task using a combination of RL and PS. Furthermore, keys and joysticks can also be used to teach the speed and shape of dynamic grasping movements as proposed by Mészáros *et al.*, 2022.

As such, we can conclude that interfaces such as keys, joysticks, and 6 DoF mouses can be used not only for their commodity value but also for allowing IIL methods to be applied to systems that cannot accept other types of interactions (e.g., kinesthetic). Furthermore, it is a common practice to implement modules that convert other types of sensors into discrete or analog signals, which can be used by learning methods in the same fashion as the ones presented in this section.

9.1.3 Contact Free Interfaces

The previous sections presented interfaces whose raw output is compatible with the type of information expected by the learning method. However, interfaces such as video or voice require more complex processing for mapping from the communication channel to the learning method's input; also, such interfaces are interesting in the IIL context to enable natural communication between the human and the robot.

Normally, processing these signals involves different kinds of pattern recognition methods that extract the relevant information from the raw signals, which are then mapped into discrete or continuous signals. Next, we summarize works that interface raw video, motion capture, depth sensors, and voice signals with IIL methods.

Video Using videos of humans performing tasks allows the user to naturally perform a task with minimal hardware requirements, and without directly dealing with the robotic system. Nevertheless, learning from videos is not trivial given their large state spaces, making them inefficient to be trained based on human demonstrations. Furthermore, videos from humans are also susceptible to the *correspondence problem*

(Nehaniv, Dautenhahn, *et al.*, 2002), which is usually mitigated by using videos featuring only the user's hand, which is mapped to the end-effector's position.

Videos have been used in IIL through pre-trained models that extract the information of interest from the images. This idea is applied to IIL by Celemin and Ruiz-del-Solar (2019), where a hand gesture recognition module is employed to provide feedback for the COACHc framework, and by Arakawa *et al.* (2018), who use a facial expression detection mapped to pre-defined positive/negative reward values to be used with the TAMER framework.

Depth Sensors and Motion Capture Depth sensors and motion capture provide information on depth, pose estimation, and motion tracking, which is often necessary to perform dynamic tasks. Motion capture usually uses cameras or an array of inertial sensors (Hindle *et al.*, 2021) to provide tracking information of objects and the human body, to be used as a human-robot interface without directly manipulating the robot.

These systems have been successfully used to track the hand of the user, which is mapped to the robot's end-effector. As such, the user can perform a task naturally, without handling the robot, which has been shown as preferable by the users in a LfD setup (Hedlund *et al.*, 2021). Motion capture has also been used by León *et al.*, 2011 to record demonstrations of the position of objects manipulated directly by the human teacher, which allows obtaining a first policy that is later on refined interactively. For instance, a Microsoft Kinect V2 has been used by Najar *et al.* (2016) to create a discrete signal from head movements (nodding and shaking), which are mapped as positive or negative feedback to the learning agent, similarly to the pressing of a key.

The Task-Instruction-Contingency-Shaping (TICS) framework (Najar *et al.*, 2020) explores learning the mapping between movements captured with a Kinect and a discrete set of pre-defined actions. This model helps the overall training process, which in turn refines the mapping in an iterative process. This makes the system robust to different users since each user associates a different meaning to the actions. A

similar idea was previously presented by Grizou *et al.* (2013), where a robot learns to perform a task and the association of pre-defined commands identified through a voice recognition module.

Voice A direct way to create a voice interface is to use pre-established commands, simplifying its recognition, where specific voice commands are used as a feedback signal (e.g., good, or bad), or to trigger specific operation modes Kaplan *et al.*, 2002; Nicolescu and Mataric, 2003; León *et al.*, 2011. Also, Tenorio-Gonzalez *et al.* (2010) propose to parse voice commands with a language recognition module to identify a pre-established vocabulary where all words have an associated reward value which is used to compose a final evaluative feedback reward; creating an analog input (discretized according to the vocabulary) to the learner.

However, such a simplistic approach cannot capture the information richness of human speech, leading to the adaption of learning-based natural language modules. Focusing on language grounding, Blumberg *et al.*, 2002 propose an interactive click trainer which learns to associate voice commands with the actions of a simulated dog. Similarly, MacGlashan *et al.*, 2014 propose a method that grounds text commands. Voice commands have also been used in a simulated environment by Cruz *et al.* (2015), where language commands are processed by an Automatic Speech Recognition (ASR) module that interactively teaches affordances. Similarly, Krening *et al.* (2017) parse natural language sentences into *advice* or *warnings* analysing sentiment. Finally, Cruz *et al.* (2018) integrate motion capture commands and voice commands using a multi-modal estimation module to teach affordances.

9.1.4 Interactions with Multiple Interfaces

Works leveraging multiple types of human input are common in the IIL literature, which is motivated by two advantages. Firstly, each type of human-robot interaction is more adequate to capture different stages of the learning process. Secondly, different types of information are used during the learning process, which can achieve a faster or more consistent learning rate (Koert *et al.*, 2020).

As an example of the first advantage, within IIL it is common to obtain an initial policy from demonstrations, which is then refined using interactive corrections. For example, Prakash *et al.*, 2020 learns an initial policy from demonstrations in a driving simulation, which is time-consuming but only requires recorded demonstrations, then the policy is interactively refined by the user. Franzese *et al.* (2021b) initiate a policy using kinesthetic demonstrations, which is later refined through teleoperated corrections in order to learn the movement stiffness, allowing for performing force-interaction tasks without the necessity of force sensors. This idea is also applied by Celemin *et al.* (2019a), where a policy is initialized using kinesthetic demonstrations and is interactively improved using a combination of corrections and RL.

As for the second advantage, its motivation lies in the fact that each learning modality has its own shortcoming; thus, using complementary methods might allow overcoming the limitations of single interfaces. This idea has been explored by Jain *et al.* (2015), where humans provide feedback first using keys to rank policies displayed in a simulation environment, and then provide kinesthetic corrections to the trajectories; the first feedback form is easier to provide and allows teaching simpler trajectories, while the second allows teaching more difficult parts of tasks. Also, in the CEILing framework (Chisari *et al.*, 2022), an initial policy (obtained from demonstrations) is interactively improved using evaluative feedback from key presses, and corrective feedback from teleoperation (joystick); the latter allows corrections, while the former allows the user to provide feedback in difficult situations in which the user could not correct the robot.

9.2 Robot-to-Human Interfaces

Even though most works in IIL focus on how to capture information and learn from humans, some work has been paying attention to the opposite communication direction. This channel allows the robot to communicate the necessity of corrections or input, as in the case of robot-gated or active learning methods, in which the user is asked for feedback only in specific cases which improve the learning process (e.g., Sadigh *et al.*, 2017; Cui and Niekum, 2018; Brown *et al.*, 2018; Biyik

and Sadigh, 2018; Biyik *et al.*, 2020; Hoque *et al.*, 2022; Franzese *et al.*, 2021a). Additionally, Li *et al.* (2016) and Koert *et al.* (2020) show that providing the human with uncertainty and performance information about the robot's actions can improve the teacher's feedback quality, improving the learning process, i.e., the teacher concurrently learning how to teach the robot. Thus, understanding how and what information to communicate to the human can be a key to enabling non-experts to interactively teach robots.

Displays Using the computer screen is the most common way for the robot to communicate with the human. Screens are suitable for displaying simulations or behavior estimates that provide insights to the user, allowing them to preview the robot's behavior and act accordingly.

The displaying of simulations has been used by LfP methods, in which a few simulation trajectories are displayed to the user, who selects the most adequate ones according to personal preferences (Jain *et al.*, 2015). Furthermore, these methods can enable learning from multiple teachers (Wilde *et al.*, 2020).

The video interface is also part of Virtual Reality (VR) kits, which usually consist of a wearable headset that streams a camera image to the user and a joystick/controller, providing standard keys and/or analog sticks for controlling the robot. For example, DelPreto *et al.* (2020) use a VR kit to teach grasping tasks using a master-apprentice model.

Voice Using verbalization of pre-defined sentences is a straightforward way to create a communication interface from a robot to a human teacher, and it has been used in IIL by Maeda *et al.* (2017), whose active learning method asks for demonstrations of reaching objects specified a priori based on uncertainty metrics.

Nevertheless, learning what needs to be communicated with the user in IIL setups is subject to recent works. For example, Shridhar *et al.* (2020) propose to interactively learn a model for composing questions that help to disambiguate the selection of objects for manipulation tasks.

9.3 Interface Design

We should consider three main aspects to successfully complete an IIL interfacing system that connects both an agent and a human: 1) the hardware interface (discussed in this section), 2) the modalities of interaction (Section 3), and 3) also the user experience (Section 10). Interface design in IIL has been a less explored topic until today. However, works in IML already identified the main factors influencing these aspects.

First, both the needs of users and the model should be taken into account in interface design, user studies can reveal false assumptions or user patterns and difficulties, which can be used to leverage the system usability and efficiency (Amershi *et al.*, 2014).

Second, the preferences of the users on models, features, and interaction modes should be considered, e.g., to allow non-expert users to build an accurate mental model of the system, hence the best feedback possible (Chatzimparmpas *et al.*, 2020; Dudley and Kristensson, 2018; Mohseni *et al.*, 2019).

Third, excessive querying should be avoided in order to reduce the undesired cognitive load of the user (Amershi *et al.*, 2014). Instead, querying should be done to promote understanding and encourage trust in the model, helping the user to understand the source of failures and solve them.

Finally, user studies can help to develop systems in which experts and non-experts are capable of understanding what's intended from the interaction with the agent (Dudley and Kristensson, 2018). For example, Chatzimparmpas *et al.* (2020) show that users tend to better interpret ML models through data visualizations.

9.4 Discussion

Since the human is the source of information in IIL systems, special attention should be paid to the interface design for these systems. Furthermore, robot-gated methods (Section 3.2.1) can interface with the human, in order to teach them how to provide better feedback, improving the overall learning process.

Nevertheless, interfacing sensors with learning methods cannot always be done directly, especially on high-dimensional problems, such as learning from video and voice, since it is required to map this sensor information into feedback signals that have to be compatible with the already established learning methods. However, using pre-defined, or pre-trained mappings implies that the information provided by the teacher is not fully explored (as these interfaces are not adaptable), leaving open the challenge of interactively learning directly from such signals. Note that existing research outside the IIL scope focuses on learning from natural language sentences (Williams *et al.*, 2018), or on using videos of demonstrations (Yang *et al.*, 2019), and it can be used as a base for extending interactive methods. Other methods in the IIL literature focus on merging different types of signals using multi-modal learning methods, e.g., voice and video are used by Jang *et al.* (2022) as input. Such models could pave the way to create interfaces in future works.

Finally, despite the interface and modality of interaction (Section 3) being related to the information used for teaching, the human aspect has also to be taken into consideration in order to achieve an efficient interactive learning setup.

10

User Studies in IIL

A major factor contributing to effective and efficient collaboration between robots and humans is the improvement of the human experience in teaching and collaboration. The topic of this section pertains to user studies in IIL. We report central guidelines on how to design user studies in IIL, based on findings of several papers on user studies in IML and IIL specifically. This section discusses how to set up a user study and what evaluation metrics to use.

10.1 Study Setup

A user study targets comprehending user needs, behaviors, and motivations through a variety of evaluation methods such as surveys, interviews, and questionnaires, in order to improve the user experience in a certain interaction scenario. User studies require previous approval from an ethics committee. Researchers should communicate to the ethics committee any potential harm that may be caused to the participants, as well as actual or potential conflicts of interest (Kuniavsky, 2003).

To effectively set up a user study in IIL, a researcher should take into consideration system aspects as well as human aspects. Although

there is flexibility in setting up a user study, researchers should take into consideration the following aspects:

- What is the study goal?
- What are the most significant metrics to support the study goal, and how to collect data related to those metrics?
- Select participants according to the study goals. How many participants? Experts or non-experts? Do they need to get familiar with the task (s)? How to design the familiarisation phase?

These items are discussed in the following sections.

10.1.1 Study Goal

The study goal is defined according to the motivation of the study. Some user studies aim to verify if non-experts can teach complex tasks to an agent. These studies make use of different metrics. For example, (Mészáros *et al.*, 2022; Franzese *et al.*, 2021b; Pérez-Dattari *et al.*, 2020; Pérez-Dattari *et al.*, 2019) apply metrics such as success rate and time of completion to evaluate teaching efficiency in manipulation and navigation tasks. Sadigh *et al.* (2017), Bajcsy *et al.* (2017), Bajcsy *et al.* (2018), Losey *et al.* (2022), Biyik and Sadigh (2018), Jain *et al.* (2015), Chu and Thomaz (2015), and Cakmak and Thomaz (2012) use model training error and subjective evaluation metrics (user's perception of tasks and policies) to measure efficacy and efficiency in teaching successful robot behaviors.

Another goal is to compare different learning methods using non-expert users. These studies measure the learning efficiency of an IIL method and its relation to human performance. Jauhri *et al.* (2021), DelPreto *et al.* (2020), Biyik *et al.* (2020), Cui *et al.* (2019), Palan *et al.* (2019), Chisari *et al.* (2022), He *et al.* (2020), and Hoque *et al.* (2021) employ robot learning metrics (task accuracy, success rate, training time, reward maximization) as well as human performance metrics (workload assessment and model perception) to compare different learning methods.

Some studies also aim to analyze how the teaching strategy used by the users influences the results. For example, Loftin *et al.* (2016) conduct a user study to determine the rate of success of teaching behaviors such as punishment-focused, reward-focused or balanced. Vollmer and Hemion (2018) evaluate different teaching strategies such as comparative, error based, or spontaneous. These strategies are rated using ground truth data and its effect on task success or failure.

10.1.2 Participants

Human-centered evaluations require objective profiling of the participants. Users' expertise in ML, IIL and users' age and gender are aspects that may influence interaction with the learning method/task. Hence, the outcomes of user studies may vary due to the fact that different users may show differentiated interaction performance profiles, for a similar interaction method/task.

Participants' Expertise

There are two kinds of participant expertise to be considered, expertise in the task domain, and expertise in robotics and ML. Users' expertise can be categorized into high, medium, and low. For example, participants with high expertise in ML tend to make predictions faster about how the system learns compared to participants with low expertise. Therefore, a proper selection of study participants should be taken into consideration. Most of the user studies in IIL either report very low ML expertise or none (Biyik *et al.*, 2020; Palan *et al.*, 2019; Chisari *et al.*, 2022; He *et al.*, 2020; Thomaz, Breazeal, *et al.*, 2006; Franzese *et al.*, 2021b; Mészáros *et al.*, 2022; Pérez-Dattari *et al.*, 2019; Jauhri *et al.*, 2021; DelPreto *et al.*, 2020; Bajcsy *et al.*, 2018; Bajcsy *et al.*, 2017). On the other hand, Jain *et al.* (2015), Bajcsy *et al.* (2017), and Palan *et al.* (2019) require that participants have medium task domain expertise in robot manipulation tasks. Finally, high domain expertise is requested in a driving task (the participants needed to own a driving license and in some cases 8 years of experience driving cars) (Cui *et al.*, 2019; Sadigh *et al.*, 2017).

Age and Gender

Participants' age, gender, demographic categories and distribution should be taken into account to structure user studies. The most targeted age group in user studies lies between 18-37 years, with some studies comprising older participants. Gender specifications of the participants are only reported in a few studies (Bajcsy *et al.*, 2017; Palan *et al.*, 2019; Loftin *et al.*, 2016); however, no correlations are found between gender and the interactive process outcomes.

10.1.3 Number of Participants

The number of participants to be recruited for a study depends on the study goal. For example, pilot studies whose goal is to evaluate the feasibility of an approach to be used in a larger scale study include small samples, while large-scale studies include larger samples. The number of participants in a user study is also limited by the experimental setups. The setups with real robots, controllers, and interfaces require preparation, calibration, and resetting in each trail which constrains the size of a user study. We observe user studies with a real-world experimental setup to include, on average, 7-15 participants. However, simulated or grid-based high-level tasks offer the opportunity for larger-scale user studies, with, for instance, 40 (He *et al.*, 2020) or 150 (Loftin *et al.*, 2016) participants.

10.1.4 Preparation Phase

The preparation phase identifies the type and amount of training provided to participants before they interact with the system being evaluated. The aim of a preparation phase is to avoid participant interaction with the experimental setup to be influenced by confounding effects, such as lack of clarity of task requirements and task execution, or insufficient familiarity with the study interface. The training phase needs to balance the amount of information provided to participants against the potential introduction of bias towards a system, technique, or model. Participants that do not need to know the robotics hardware

details or controller interfaces are just provided with a brief session about the tasks and related high-level instructions (DelPreto *et al.*, 2020; Hoque *et al.*, 2021; Loftin *et al.*, 2016; Biyik *et al.*, 2020; He *et al.*, 2020; Pérez-Dattari *et al.*, 2019; Biyik and Sadigh, 2018; Sadigh *et al.*, 2017; Chu and Thomaz, 2015; Vollmer and Hemion, 2018). On the other hand, when participants need prior knowledge in robotics (hardware details, controller interface), a more dedicated session for familiarization is scheduled before the main study (Mészáros *et al.*, 2022; Franzese *et al.*, 2021b; Chisari *et al.*, 2022; Bajcsy *et al.*, 2017; Bajcsy *et al.*, 2018; Palan *et al.*, 2019; Jain *et al.*, 2015).

10.2 Evaluation Methods

In this Section, we describe evaluation methods and metrics that can be adopted in IIL. These include, *robot learning performance metrics* and *human performance metrics*.

10.2.1 Robot Learning Performance Metrics

IIL makes use of quantitative metrics to evaluate robot learning performance. Deciding on how to make use of these metrics depends on the goal and objective of the learning model. Common metrics to evaluate robot learning performance in IIL are listed below.

Cumulative Reward

A widely used metric to evaluate robot task performance in IIL is the cumulative reward during task execution. In the IIL case, the reward function would need to be additionally designed by the creator of the user study for a given task, for evaluation purposes. This reward is also communicated to the study participants so that they are consistent with the true reward for the task during their interactions. Examples of tasks that make use of the cumulative reward metric include manipulation reaching task (Cheng *et al.*, 2018; Pérez-Dattari *et al.*, 2019; Pérez-Dattari *et al.*, 2020), writing symbols (Schroecker *et al.*, 2016; Jauhri *et al.*, 2021), autonomous driving (Menda *et al.*, 2017), UAV racing

task (Li *et al.*, 2019b), balancing task on Cartpole (Wilson *et al.*, 2012; Knox and Stone, 2012; Vien and Ertel, 2012), games like atari, pacman, frogger (Christiano *et al.*, 2017; Cederborg *et al.*, 2015; Subramanian *et al.*, 2016), and OpenAI Mujoco based locomotions (Reddy *et al.*, 2019; Hoque *et al.*, 2021; Cronrath *et al.*, 2018).

Success Rate

Success rate evaluates the robustness of a learning method in performing a task. This metric is defined by calculating the ratio between the number of successful executions of the task and the total number of attempts. This metric is more useful for tasks in which success is binary (i.e, completed or not) and it requires low design effort (in contrast to the reward function). It is calculated among several trails/episodes and it is used for different tasks such as grasping (DelPreto *et al.*, 2020), contact-rich manipulation (Mandlekar *et al.*, 2020; Chisari *et al.*, 2022; Ablett *et al.*, 2020), reaching manipulation (Pérez-Dattari *et al.*, 2020), autonomous driving (Cui *et al.*, 2019; Prakash *et al.*, 2020), mobile robot navigating to a target (Argall *et al.*, 2008), drone navigation to reach a goal with tolerance (Blukis *et al.*, 2018), drone perching (Goecks *et al.*, 2019), game of maze (Le *et al.*, 2018), and industrial manipulation (Luo *et al.*, 2021; Hoque *et al.*, 2022).

Model Training Error

This metric measures the accuracy of a model in fitting the observed data, and it is useful to evaluate or debug the model, and also observe how good it resembles the data. This metric is used in tasks such as autonomous driving (Biyik and Sadigh, 2018; Sadigh *et al.*, 2017), walking (Akrour *et al.*, 2014), and reaching (Bajcsy *et al.*, 2018; Losey *et al.*, 2022; Biyik *et al.*, 2020; Palan *et al.*, 2019). Furthermore, this metric is also used for performance evaluation in reconstructing trajectories for a task of drawing symbols (Celemin *et al.*, 2019a), and driving a mobile robot (Spencer *et al.*, 2020; Kelly *et al.*, 2019).

Task Completion Time

This metric represents the total time taken by the algorithm to learn to solve the task. Examples on studies that have used this metric include item sorting, where human guidance is used to help the agent learn the task quickly (Suay and Chernova, 2011), mobile robot navigation task (Tenorio-Gonzalez *et al.*, 2010; MacGlashan *et al.*, 2017), game of soccer (Meriçli *et al.*, 2010), insertion time in industrial assembly tasks (Luo *et al.*, 2021), high-level grid-based tasks (Peng *et al.*, 2016), and constrained manipulation (Hoque *et al.*, 2022). Some authors use switching time to evaluate the learning algorithm performance (Hoque *et al.*, 2021).

Safety Performance

The safety of an agent depends on the combined performance of the robot and the human, e.g. when learning, an autonomous driving car may drive off the road. This metric can be measured by the uncertainty of the agent's actions. The higher the uncertainty, the lesser the safety. Menda *et al.* (2017) and Menda *et al.* (2019) use safety performance evaluations by calculating the average reward resulting from the combined robot and human actions.

Task Specific Metrics

Specific metrics evaluate task performance in a certain domain. In a manipulation task, *number of knocked items* is an effective way to measure the accuracy of the task by observing how many objects other than the target object are displaced or knocked over by the agent/robot, e.g., in a constrained manipulation task with cluttered objects (Laskey *et al.*, 2016). In an autonomous driving application, *infraction rate* is a leading performance metric. Infractions refer to breaking the driving rules of traffic (e.g., collisions, intersections with the opposite lane, driving onto the curb etc.), leading to potential collisions and human injuries. Cui *et al.* (2019) and Prakash *et al.* (2020) used infraction rate metrics to derive the quality of autonomous

driving performance. The distance traveled by the autonomous vehicle between infractions has been used as another performance indicator also (Cui *et al.*, 2019; Kahn *et al.*, 2021). For path following tasks, *mean path deviation* measures the average error of the robot over multiple trials, e.g. lane deviations in autonomous driving. Bajcsy *et al.* (2018) use this metric in a task of object placement with human preference. The evaluation focuses on observing whether the robot follows the human preferred path and the mean deviation resultant from it. Kelly *et al.* (2019) and Ross *et al.* (2011) measure the mean deviation from the lane per meter/lap in autonomous driving. Bootsma *et al.* (2021) measure the root mean square error of deviation from the desired path for mobile robot navigation.

10.2.2 Human Performance Metrics

Human Performance Metrics evaluate different human performance capabilities (Abdel-Malek *et al.*, 2005). According to Sperrle *et al.* (2021), there is no established methodology to evaluate HCML systems due to the field's novelty. The authors emphasize that human factors such as effort and trust (cognitive and emotional elements), as well as placing humans as actors in the design of ML models need to be taken into consideration in order to improve evaluations of the HCML process.

Unfortunately, there are no methods in the literature for evaluating every feedback signal provided by the teacher, since different sequences of signals can obtain the right effect on the learned model. Therefore, the only way to evaluate whether the given feedback is correct is by the testing of the policies.

Below we describe some of the Human Performance Metrics applied in IIL, like human workload and users' perception of robot model/behaviors. According to the literature, the previous metrics evaluate different aspects of human performance in HCML (Cui *et al.*, 2021; Kulesza *et al.*, 2014; Mohseni *et al.*, 2019); hence, they are presented in two different sections. The aforementioned metrics should be combined with the Robot Learning Performance metrics to better evaluate and improve the outcomes of the HCML process.

Number of Interactions/Interventions from a Teacher

One of the main goals related to IIL is to minimize human effort during the teaching experience. The amount of feedback provided by the teacher for successful task execution is a quantitative measure of human effort — the amount of feedback tends to correlate with the effort levels. This metric is used in IIL to evaluate human effort during the teaching experience (Bajcsy *et al.*, 2018; Bootsma *et al.*, 2021; Celemin and Ruiz-del-Solar, 2019; Cronrath *et al.*, 2018; DelPreto *et al.*, 2020; Franzese *et al.*, 2021b; Goecks *et al.*, 2019; Hoque *et al.*, 2021; Jain *et al.*, 2015; Laskey *et al.*, 2016; Le *et al.*, 2018; Myers *et al.*, 2022; Najar *et al.*, 2016; Peng *et al.*, 2016; Tenorio-Gonzalez *et al.*, 2010)

Interaction Time

Interaction time is a metric that allows measuring human effort in IIL tasks. Commonly, it includes the total study duration for a teacher to successfully train a model (Chisari *et al.*, 2022; Franzese *et al.*, 2021b; Pérez-Dattari *et al.*, 2019; Losey *et al.*, 2022; Bajcsy *et al.*, 2018; Jain *et al.*, 2015). Hoque *et al.* (2021) also consider the time for switching between interactive mode and autonomous robot mode.

Teacher's Cognitive Load and Engagement Levels

This metric represents the cognitive load levels and engagement levels of the teacher during the learning cycle, which can be measured via the NASA Task Load Index (NASA-TLX), Van der Laan questionnaires (Van Der Laan *et al.*, 1997) for perceived workload (Mészáros *et al.*, 2022; DelPreto *et al.*, 2020; Jauhri *et al.*, 2021; Hoque *et al.*, 2022), and/or via physiological measures (Ferraz *et al.*, 2019; Cui *et al.*, 2021). The NASA-TLX is a widely used subjective assessment tool that rates human perceived workload and performance for a certain task. The total workload is divided into six subjective subscales comprising i) Mental Demand, ii) Physical Demand, iii) Temporal Demand, iv) Performance, v) Effort and vi) Frustration. Subjective measures of cognitive load such as the NASA-TLX tend to have a high variance across humans (Cui

et al., 2021). Additionally, it can be helpful to make use of objective measures, e.g., physiological metrics such as electroencephalographic analysis and physical exertion monitoring, both recently used in human-robot interaction to objectively quantify human cognitive load and physical load. Increases in human workload correlate with a decrease in task performance in general human-robot interaction tasks (Ferraz *et al.*, 2019).

User Perception of Robot's Behavior

The *user mental model* is measured by asking users their view on the logic behind the model decision-making process, i.e. users' perception of how correctly the robot learns the task. In a robotic task, the user observes the execution of the robot and evaluates its behavior. Evaluation methods include interviews, think-alouds, self-explanations, Likert-scale questionnaires, users' model output prediction, and users' model failure prediction (Biyik *et al.*, 2020; Palan *et al.*, 2019; Bajcsy *et al.*, 2018; Losey *et al.*, 2022). Sadigh *et al.* (2017) evaluate users' perception of whether the robot understands and executes the driving task as desired by the users. Jain *et al.* (2015) measure users' perception on validity/correctness of robot motions.

User Trust and Reliance

User trust and reliance represents ratings for whether the users trust the robot or not, e.g., to operate safely in a factory, office, or household environment. Evaluation methods include subjective measures, e.g. interviews, self-explanations and/or Likert-scale questionnaires; and objective measures, e.g. user perception of model competence to execute a task (Bajcsy *et al.*, 2018; DelPreto *et al.*, 2020). DelPreto *et al.* (2020) ask users to rate (on a scale from 1-7) how likely they would trust a robot in a variety of settings and tasks in the context of grasping items - whether they would trust the robot to pick up different objects, work with power tools, or operate in a classroom.

Easiness of Interactions and Usability

Easiness on interactions and usability represents users' ease and confidence when interacting with a robot in order to complete a task. This metric can be measured by subjectively asking questions to users (Bajcsy *et al.*, 2017; Bajcsy *et al.*, 2018; Biyik *et al.*, 2020; Cui *et al.*, 2019; Franzese *et al.*, 2021b; Jain *et al.*, 2015; Palan *et al.*, 2019). Cui *et al.* (2021) report that several researchers have been using the System Usability Scale to measure the perceived usability of a ML system — this scale evaluates users' ease and confidence when using a system. The System Acceptance Scale can also be used in ML to evaluate the acceptance of new technologies due to its simplicity and reliability (Van Der Laan *et al.*, 1997).

10.3 Discussion

Even though there have been a few improvements in user studies in IIL, many aspects have not yet been covered in the literature, e.g. objective measures of human performance as well as evaluation of the users' mental model. Nevertheless, they are necessary to perform more accurate evaluations on human response to interactions with artificial agents and to improve overall efficacy and efficiency in the human-agent interaction process.

A systematization on how to design user studies in IIL is also necessary to better inform researchers on how to carry them in this field. User studies in IIL should target clear communication between a human and an artificial agent, in addition to a user-friendly, intuitive, and enjoyable experience for the human supervisor.

11

Benchmarks and Applications

Evaluating robotic systems is a well-known and difficult challenge due to the wide variety of robots, tasks, and environments tailored for each robotic system (Behnke, 2006). Such difficulties are exacerbated in the context of Human in the Loop (HIL) learning, where the performance of learning methods is highly influenced by the data used (Bouthillier *et al.*, 2021), which often comes from humans. Furthermore, the different modalities of interactions (see Section 3) influence the quality and amount of information that humans provide as feedback during learning, creating the necessity to compare not only the learner's performance but also the human aspect of an IIL system (see Section 10).

The lack of reproducibility of experiments has been debated for years in the field of cognitive robotics. In order to leverage the opinions of experts on how to evaluate such systems, Aly *et al.* (2017) present the discussion entitled *"Towards Intelligent Social Robots - Current Advances in Cognitive Robotics"*.[1] They recognize the difficulties in testing and comparing complex robotic systems in a fair manner, as well as a lack of proper methods and tools to do so.

[1] Held at the *"15th IEEE-RAS Humanoids Conference, Seoul, South Korea, 2015"*, https://intelligent-robots-ws.ensta-paristech.fr/

A common way to compare IIL methods is to evaluate their performance on specific applications, which can be tailored to demonstrate specific aspects and contributions of learning methods or to achieve task completion. A different approach is to leverage simulation and datasets to achieve some degree of uniformity. Sections 11.1, 11.2, and 11.3 present the main applications, datasets, and applications used for evaluating and comparing IIL approaches, respectively.

11.1 Applications

During the research and development of IIL methods, a vast number of applications have been proposed which can potentially benefit from the intrinsic characteristics of this field. On one hand, such applications can be designed to validate or demonstrate research ideas, showing gaps in previous methods and how they are addressed by the proposed one, or to evaluate the learner's capabilities, such as learning rate or generalization. These applications are called *testbed applications*. On the other hand, applications can target the consumer or products, whose requirements and preferences in a specific task are the target of evaluation. These applications are called *use-case applications*.

11.1.1 Testbed Applications

We categorize the testbed applications into their task domains (e.g., robot manipulation or robot navigation), identifying representative tasks and successful demonstrations of IIL methods.

Manipulation

Pick and Place This task refers to picking an item in one location, moving it, and placing the item in another location. It is a fundamental automation task that is commonly used as a sub-task in other long-horizon manipulation tasks. Because of this, it is commonly found in a variety of robotics applications.

Human-like picking is a challenging robotic task that requires learning the constraints and adaptations for non-zero-velocity end-effector

movements, orientations, and gripper width precisely for a successful pickup; otherwise, the task will probably fail. An IIL approach is suitable to learn this task efficiently by improving an initial demonstration through interactive corrections. Mészáros *et al.* (2022) applied an IIL approach for the task of non-zero velocity picking of objects using a 7 DoF Panda robot. The users could overcome the limitations they had during the demonstrations and teach the desired behaviors using corrections.

Additionally, Bajcsy *et al.* (2017), Bajcsy *et al.* (2018), and Losey *et al.* (2022) develop an online physical human-robot-interaction method to learn successful object placement tasks with user preferences. Figure 11.1 shows the experimental validation by Losey *et al.* (2022) where the robot learns object placement with human preferences.

Contact-rich Manipulations Contact-rich manipulation tasks are space-critical tasks, meaning that certain regions of the state space require precise sequences of actions to make meaningful progress. These regions are a bottleneck for successful task execution because a small deviation from the correct policy may lead to failure, e.g., the *peg-in-hole* task. Task failures due to inaccuracies in the agent's actions while traversing critical state-space regions can be avoided using IIL.

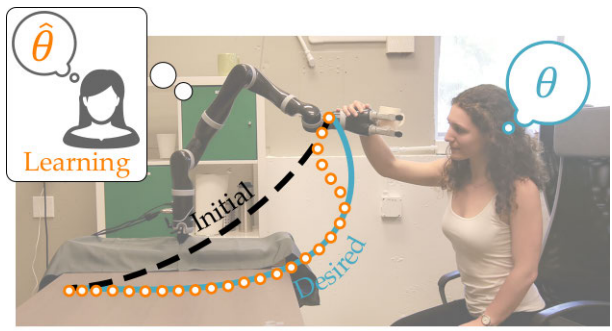
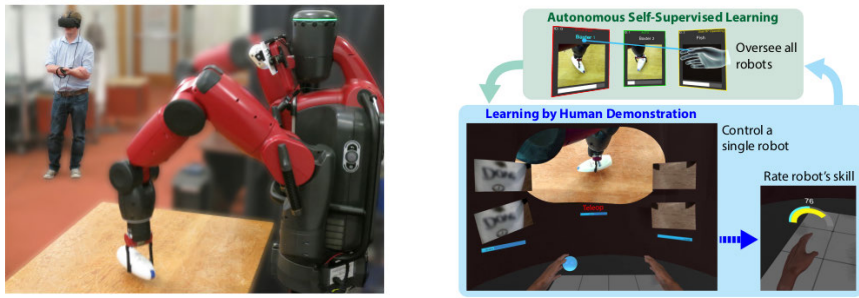


Figure 11.1: A robot learns from physical human interaction to understand the task objective (i.e., go closer to the table). Source: (Losey *et al.*, 2022).

One of the methodologies to traverse such bottlenecks is to learn a policy by requesting feedback from an expert when the agent is not able

to find solutions. DelPreto *et al.* (2020) apply an IIL approach where the policy predicts a vector of confidence scores for four different gripper orientations, and the one with the highest confidence is selected. The robot autonomously attempts a predicted grasp and detects whether it can lift the object and hold it for a fixed amount of time. If the robot repeatedly fails to execute a successful grasp orientation, it requests user assistance, as shown in Figure 11.2.



(a) **(b)**

Figure 11.2: Figure (a) shows the setup where a person supervises a robot for a grasping task. (b) shows the virtual reality interface where the user remotely gives grasping demonstrations. Source: (DelPreto *et al.*, 2020).

Instead of agents requesting help, another way of traversing the bottleneck in these tasks is learning based on human-gated interventions. The expert supervises the agent's task execution and provides intervention when considered necessary. Mandlekar *et al.* (2020) use this approach in two contact-rich manipulation tasks for a 6-DoF robot manipulator: a threading task and a coffee machine task. The threading task consists of the robot carefully grasping a rod and inserting it into a hole. The coffee machine task consists of the robot grasping a coffee pod, inserting it into the coffee machine, and closing the lid. These tasks contain critical regions while performing sub-tasks like grasping, inserting, and closing.

The presented methods assume that the expert is always able to correctly intervene when necessary. However, this might not be possible with non-expert users. For such a scenario, Chisari *et al.* (2022) combines

human interventions with evaluative feedback. The user intervenes to correct undesirable behavior. The evaluative feedback is used to select or discard the part of the trajectory which the user cannot correct. The approach is applied to learn contact manipulation tasks: pushing a box, picking up a cube, and pulling a plug from real-world high-dimensional image observations using a real KUKA robot manipulator. This approach shows that the combined feedback-based strategy is more advantageous than BC (Osa *et al.*, 2018) and different interactive learning strategies (Kelly *et al.*, 2019; Mandlekar *et al.*, 2020).

Finally, Franzese *et al.* (2021b) employ demonstrations and corrections on desired end-effector transitions, to learn policies able to perform well during critical regions traversal. The approach is successfully validated in tasks of plug removal/insertion, pushing box and wiping whiteboard using a 7 DoF Panda robot manipulator. The policies also learn less stiff control behavior in free regions which is desirable for safe manipulations in human-robot environments.

Ball in a Cup This is a challenging robotic manipulation task where there is a cup attached to the robot's end-effector and a ball attached to the cup by a thread. The goal is to move the end-effector to swing the ball making it land in the cup. This task is a challenging manipulation benchmark in robotics, as it corresponds to an underactuated system that depends on hard-to-model physical factors. This task has been approached using an IL based PS optimization on a real 7 DoF Barrett WAM robot manipulator (Kober and Peters, 2008).

Later, Celemin *et al.* (2019a) apply an IIL method for learning the ball-in-a-cup task by combining human teacher's knowledge during the PS exploration process. The teacher provides relative corrections to the robot's end-effector. Experiments are carried out using a PS method on the experimental setup shown in Figure 11.3. Results show that the human interactions lead to an improvement in convergence speed of the PS method by an order of 4 to 8.

Writing Symbols The task of writing symbols consists of generating an accurate end-effector trajectory with a particular shape. Unfortunately,

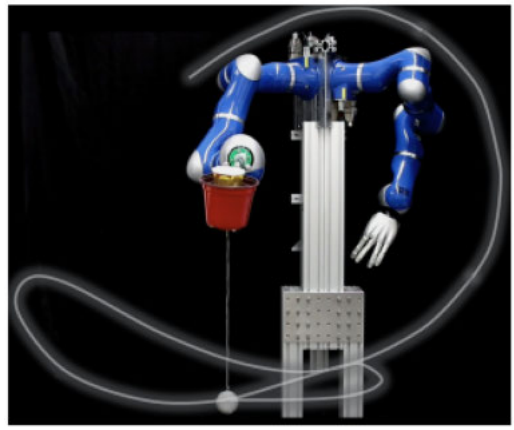


Figure 11.3: Ball-in-a-cup set up used in Celemin *et al.* (2019a) composed by a 7DoF KUKA lightweight arm and an OptiTrack system, which tracks the positions of the ball and the cup.

a combination of improper user interfaces and insufficient user expertise may limit obtaining high-performance writing demonstrations.

In absence of good initial demonstrations, Schroecker *et al.* (2016) develop an IIL method that uses soft via-points based demonstrations to initialize a writing policy and interactively refine it through a PS process. This approach is validated for the task of writing symbols in simulations. The results show accurate and smooth symbol reproduction without having good quality human demonstrations of the task.

Celemin *et al.* (2019a) propose a mechanism for writing tasks using interactive corrections during robot execution. This approach is validated for writing symbols using a real 6 DoF robot arm. The results show that a reduction of 84.4% in symbol reproduction error is achieved using the interactive PS method in comparison to a non-interactive one.

Item Sorting The task of item sorting consists of ordering items based on their categories. Suay and Chernova (2011) apply an IIL approach for using human evaluative feedback and human guidance in a RL setting for an object sorting task. The item sorting robotic task setup, as seen in Figure 11.4, consists of an Aldebaran NAO humanoid robot in front of a table with different objects placed in three zones, along

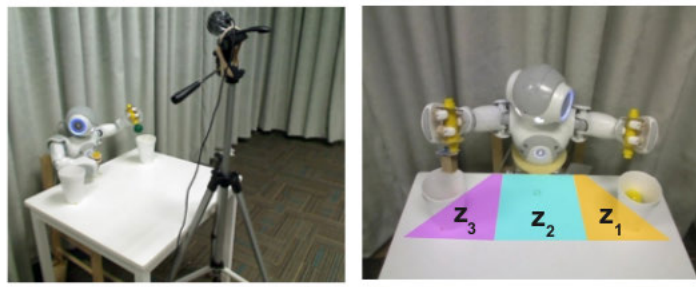


Figure 11.4: Experimental setup for object sorting using human guidance. Source: (Suay and Chernova, 2011).

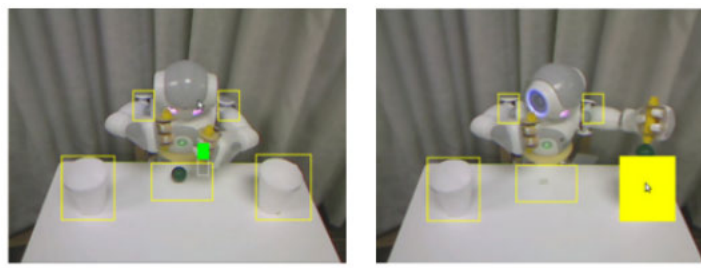


Figure 11.5: Interface for human guidance. Figure shows (left) how to give a positive reward by left click and mouse drag upwards; (right) shows how a user guides the robot by selecting a region of interest out of five rectangular regions, each associated with an action set. Source: (Suay and Chernova, 2011).

with storage bins for each category. The human teacher interacts with the robot through a graphical user interface shown in Figure 11.5. This study shows that the robot learns to use its camera to identify the characteristics of the objects and pick them up and place them in the appropriate cup.

The task of item sorting in a dynamic environment, e.g., on a conveyor belt, requires time-critical movements of the robot to grasp the moving items and sort them as per their category. A task of fruit sorting on a moving conveyor is shown using IIL by Pérez-Dattari *et al.* (2020). The setup consists of a conveyor belt transporting oranges and pears, a 3 DoF robot arm, and a head-mounted camera. The robot uses raw camera images as input and selects oranges with its end-effector and moves away from the pears.

Navigation

Autonomous Driving Autonomous driving is a challenging robotics problem due to varying environments, terrain, topologies and dynamic vehicle/pedestrian interaction or disturbances. This area of research aims to develop intelligent autonomous vehicles to achieve safe and high-performance mobility (Pérez-Dattari *et al.*, 2022). The problem consists of steering the vehicle based on sensory observations (camera, radar, lidar etc.) to weave around the obstacles and navigate on the road.

Kelly *et al.* (2019) allow human experts to take control when they deem it necessary, and to maintain exclusive control authority until they manually hand control back to the agent. The approach is experimentally validated in driving a vehicle without collisions along a road with other stationary vehicles as obstacles. The vehicle and testing setup are shown in Figure 11.6.

Figure 11.6: Test vehicle (left) and expert driver interface (right). Source: (Kelly *et al.*, 2019).

In addition to safe driving performance, it is also important to minimize the expert's burden during the interaction. Hence, a technique for querying the human expert is presented in Cui *et al.* (2019). In this approach, the agent predicts uncertainty to anticipate risky states for the vehicle and switches control to the human expert to prevent dangerous situations. The experimental results from simulated driving tasks in CARLA driving simulation environment (Dosovitskiy *et al.*,

2017) demonstrate that the uncertainty estimation method can be leveraged to reliably predict risky states and minimize human efforts.

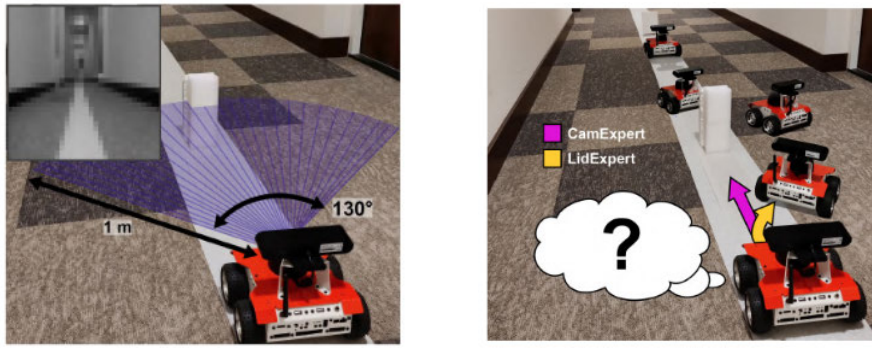
Mobile Robots To improve the performance and efficient exploration of mobile robot applications, IIL approaches are explored in the literature. Knox *et al.* (2013) demonstrate the usefulness of the TAMER framework for interactive navigational behaviors on a real mobile-dexterous-social robot platform called Nexi. The task consists of navigation to a marker that can be moved by the human trainer. The experiments show that a robot can learn to perform sequential navigation tasks using only real-valued feedback on its behavior from a human trainer.

Pérez-Dattari *et al.* (2018) show an application for learning to drive a Duckiebot autonomously through the desired track from the project Duckietown (Paull *et al.*, 2017). The robot uses raw visual information from an onboard camera to follow a path without leaving the road. The human teacher observes the task and advice linear and angular velocity corrections to the robot's actions using a keyboard in case of deviations. The robot is able to learn the task of navigation from scratch using only human corrections via the D-COACH framework in approximately 6 minutes.

Bootsma *et al.* (2021) show learning navigation behavior for an autonomous mobile robot by leveraging the strengths of different sensors, under human supervision. They demonstrate that the learning method can prevent dangerous navigation behaviors. The experiment was carried out on a real mobile robot ROSBot, equipped with a 2D lidar and a camera. The experimental setup is shown in Figure 11.7.

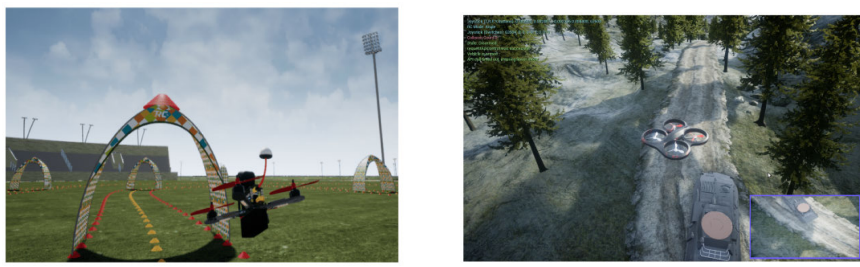
Drones Drone navigation consists of successfully maneuvering to reach a target while avoiding obstacles. Drone navigation is a challenging task since it can include non-linear dynamics, blurry images from the moving drone, and the presence of environmental gust disturbances.

The approach introduced by Li *et al.* (2019b) enables learning from multiple non-expert teachers by discarding bad drone maneuvers. The capabilities of this approach are demonstrated in drone navigation through racing tracks in the Sim4CV racing environment (Müller *et al.*, 2018) shown in Figure 11.8(a).



(a) (b)

Figure 11.7: Learning navigation behavior for an autonomous mobile robot. The figure shows conflicting decisions based on two different sensory feedback. In conflicting situations, an expert is queried to take control and the correct action of a sensor-fused policy. Source: (Bootsma *et al.*, 2021).



(a) (b)

Figure 11.8: Figure shows (a) Drone Navigation and (b) Drone Perching. Source: (Li *et al.*, 2019b), (Goecks *et al.*, 2019).

Moreover, Goecks *et al.* (2019) demonstrate an aerial robotic perching task using a drone in AirSim simulator (Shah *et al.*, 2018). The simulated setup in Figure 11.8(b) shows the drone with a downward-facing camera hovering over the moving landing platform.

11.1.2 Use-case Applications

Use-case applications consist of applications for a targeted audience. IIL research can be employed to provide targeted solutions to specific problems by considering domain knowledge.

Assistive Robots

The Healthcare industry offers opportunities for many important use-cases of an assistive robot e.g., feeding, dressing and shoe fitting for disabled people. A possible way to empower such attributes is through IIL. For instance, Canal *et al.* (2021), Canal *et al.* (2018), and Canal *et al.* (2016) develop a robot personalization framework for three different assistive applications i.e., feeding, shoe fitting and dressing, where the robot performs each task in a different manner based on corrective feedback from the user. The experimental setups for different applications are shown in Figure 11.9.

(a) User feeding **(b)** Shoe fitting **(c)** Dressing garment

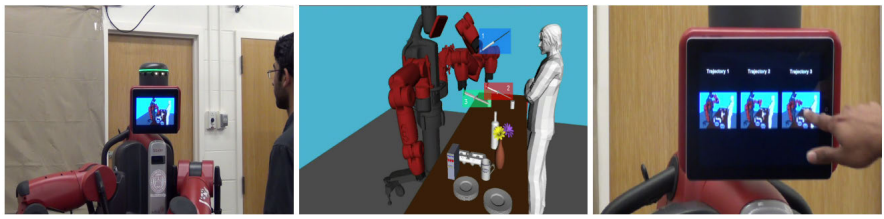
Figure 11.9: Assistive personalized application. Source: (Canal *et al.*, 2021; Canal *et al.*, 2016).

Household Robots

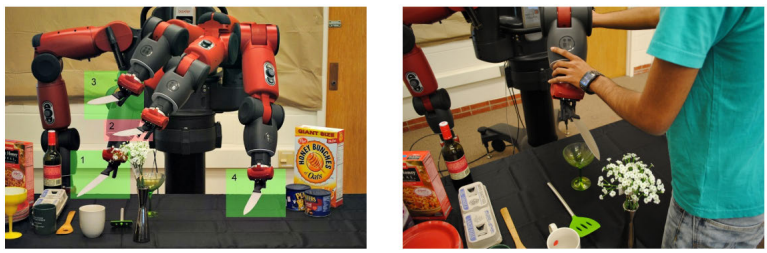
Robots are expected to serve alongside humans in household environments. In these environments, the successful execution of the task depends not only on accurate robot motions, but also on safety measures (not damaging fragile items or not hurting nearby humans) and on a clear understanding of human preferences. Although complex behaviors

like robot motions can be taught using human demonstrations, safe behavior and human preferences are difficult to anticipate and model.

Jain *et al.* (2015) demonstrate a case study using PR2 and Baxter robots working in human-centered environments for tasks like household chores and grocery checkouts, as shown in Figure 11.10. They propose an IIL framework to incorporate user feedback and improve its result as per user preferences. The approach is experimentally validated on 35 robotic tasks in household settings.



(a) Ranking-based preference feedback: (Left) Robot ranking of trajectories and (Middle) displays top three trajectories on a touch screen device (iPad here). (Right) User selects the trajectory as per his preference.



(b) Demonstration-based preference feedback. (Left) robot displays the planned trajectory (waypoints 1-2-4) and human corrects waypoint 2 because it is very near to flower (Right) demonstration on waypoint 2

Figure 11.10: Learning user preferences for household manipulation tasks. Source: (Jain *et al.*, 2015).

While working in household environments, a robot has to encounter multiple user preferences for the same task. Learning a unimodal reward from data with inconsistent preferences, coming from multiple users, is likely to result in a low-quality policy. To address successful task execution in such situations, Myers *et al.* (2022) validate an active query-based IIL method for learning multimodal reward functions from

multiple human preferences. The approach is evaluated with a Fetch robot on the task of learning to shelve an item using multiple users' feedback (Figure 11.11).

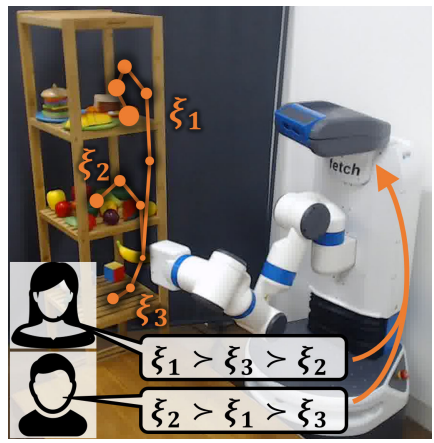


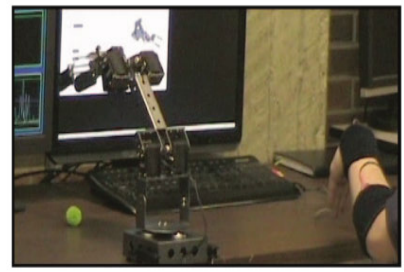
Figure 11.11: Figure shows the robot taking a decision based on multiple users' preferences to shelve an item. Source: (Myers *et al.*, 2022).

Medical Robots

Surgical Needle Insertion In a medical surgery, a robotic assistant is expected to help the surgeons, for instance, in suture tying. Such tasks require an accurate and precise insertion of the surgical needle, where inaccuracies could lead to an undesirable outcome, such as failed suture or wounded neighboring tissues. For such applications, Laskey *et al.* (2016) and Hoque *et al.* (2022) validate IIL on a Surgical Da Vinci Research Kit Figure 11.12(a).

Myoelectric Prostheses Multi-functional myoelectric prostheses are used to control the movement of robotic appendages. To do so, they monitor electrical signals from muscle tissues, which are produced from the deficient limb. In these applications, the patient's intent and usage pattern are very important to consider. Therefore, clinical and technical intervention is always required for a patient to improve the device's performance. Therefore, to develop artificial intelligent limbs for patient

(a) Suture Tying on a Surgical Da Vinci Research Kit. Source: (Laskey *et al.*, 2016).



(b) Prosthesis Limb Controller for AX-12 Smart Arm. Source: (Pilarski *et al.*, 2011).

Figure 11.12: Assistive robots in medical applications.

customization, Pilarski *et al.* (2011) develop successful limb controllers which are initialized using prostheses data and improved using human-delivered signals. The feedback-based learning framework is validated using AX-12 Smart Arm (see Figure 11.12(b)) which shows that method can be adapted to varied application settings and patient needs.

Industrial Robots

Industrial applications are composed of precision tasks in constrained environments, which are commonly addressed using handcrafted solutions. Hence, as soon as the object of manipulation and the surroundings vary, a significant amount of time and expertise is needed to adapt the solutions. IIL approaches are very useful in such settings, as they allow to learn the task from a few examples and generalize to similar tasks. Luo *et al.* (2021) introduce a framework for learning robust robotic manipulation policies in industrial settings by leveraging demonstrations and human corrections. In this approach, human demonstrations of the industrial tasks are collected using teleoperation. An initial policy is learned from this data and corrected during run-time by a human teacher. The method is evaluated on three challenging tasks of insertion in static and dynamic settings as shown in Figure 11.13.

(a) NIST board insertion **(b)** Moving HDMI Insertion **(c)** Key-lock insertion

Figure 11.13: Robot in Assembly Tasks. Source: (Luo *et al.*, 2021).

11.2 Datasets

Datasets are an important resource for developing and testing computational methods. They aim to provide data with the same characteristics that are expected in real environments, but that can be processed offline. However, it is challenging to use datasets for directly testing IIL methods, given that, in many cases, they can only be provided by an online source, the human teacher. For instance, it is common practice to use datasets for policy initialization during experiments, enabling a warm-start which can lead to better policies or shorter training sessions, which is interesting for the human teacher. However, using datasets for policy initialization is limited for setups and tasks similar or equal to the one used to collect the data, preventing their broad adoption in IIL.

RoboTurk RoboTurk (Mandlekar *et al.*, 2018; Mandlekar *et al.*, 2019) presents a large dataset of manipulation tasks with robotic arms, whose novelty lies within its crowd-sourced model, i.e., demonstrations from many people were collected and saved in the database. During the demonstrations, the robots are controlled using a cellphone, which is a widely available interface and it does not compromise the learning performance. The user receives a live stream from the camera of the robot set-up (or its simulation). Current developments on RoboTurk aim to enable humans to perform corrective interventions also using the mobile phone and video stream interface (Mandlekar *et al.*, 2020), effectively enabling HIL learning.

RoboNet Similar efforts in collecting a large number of demonstrations are performed by Dasari *et al.* (2020), who presents a collection of demonstrations of manipulation tasks using a diversity of robot arms, multiple view-point cameras, and objects. The dataset is proposed to be used to learn an initial policy, which can be refined by interactive approaches, leading to improved learned performance. Furthermore, the variance in the data collection is considered a feature, given the argument that learning methods should be able to generalize across different setups, reducing the limitation on how similar the tasks and applications have to be for benefiting from dataset-based policy initialization.

11.3 Benchmarks

Traditional benchmarks aim to provide an accurate performance metric that allows comparisons (Fleming and Wallace, 1986) in a standardized manner. Benchmarks are often constructed by carefully selecting standard data and metric measures for a specific set of applications; otherwise, variations in the results might render analysis insignificant (Curnow and Wichmann, 1976).

The necessity of standardization makes sense in traditional computer systems. However, not much work has been done towards the creation of benchmarks for IIL specifically, since the human teacher is the main source of the data, and its modeling remains challenging. Therefore, given the lack of options, several IIL papers have evaluated their methods using RL benchmarks. Although useful to evaluate important aspects of the learning methods, they are not able to provide reproducibility when it comes to evaluating the human factor of IIL. Consequently, in the following of this section, several benchmarks that were not necessarily designed for IIL are analyzed with the purpose of providing ideas, or starting points, for future work.

OpenAI Gym One of the most popular benchmarks for testing IIL approaches is the OpenAI Gym (Brockman *et al.*, 2016), which contains Atari games, and simulator-based tasks for classical control and for robotics, simulated using the MuJoCo physics engine (Todorov *et al.*, 2012). Examples of works tested on OepnAI Gym are Akrour *et al.*,

2012; Hoque *et al.*, 2021; Palan *et al.*, 2019; Menda *et al.*, 2019; Myers *et al.*, 2022.

Surreal Surreal (Fan *et al.*, 2018) is a simulated framework for RL methods which includes a benchmark for robot arm manipulation tasks. It provides RGB camera, depth map, and proprioception to be used by the learning methods. Furthermore, teleoperation is supported with $3D$ motion capture devices (VR controller), which can be used to provide demonstrations or feedback. The evaluation is performed through simulation of manual and bi-manual setups of Saywer robots using the MuJoCo (Todorov *et al.*, 2012) simulation environment. Moreover, surreal is used by Lee *et al.* (2020), who perform a state selection step for learning object manipulation tasks from human demonstrations.

Habitat Habitat (Savva *et al.*, 2019) is a simulation-based platform for training embodied agents (virtual robots), which organizes datasets, simulators, and tasks in different layers, aiming to enable transfer learning among different robotic agents. Habitat 2.0 (Szot *et al.*, 2021) expands the previous work with *home assistants*, including a $3D$ dataset of apartments with articulated objects for navigation and high-level robot manipulation tasks (e.g., tidy the house, stock groceries, set the table). The environment and evaluation are all based on Bullet physics simulation (Coumans, 2015), enabling to perform simulations 2 orders of magnitude faster than other similar simulations. Towards benchmarking, the learning methods are provided with RGB, depth, GPS, and Compass inputs and are evaluated in high-level tasks such as *tidy the house*, *Prepare groceries*, and *Set the table*. The habitat platform is used by Lin *et al.* (2022), where graph NNs is used to learn manipulation tasks in a LfD setup.

Meta-World Meta-World (Yu *et al.*, 2019) is a simulation-based benchmark focused on meta-reinforcement learning and multi-task learning. It provides a broader distribution of the tasks in contrast to what is previously used by the meta-reinforcement learning community. It emphasizes generalization to distinctly new tasks, beyond plane parametric

variation in a single (or a limited number of) tasks by providing 50 distinct robotic manipulation tasks (e.g., pick-place, reach, push, open window, open cabinet). Although all provided tasks are short-horizon (not a long-horizon sequence of tasks) it provides a valuable benchmark and is also used in the robot learning community (Sinha *et al.*, 2022). Meta-World tasks are implemented in the MuJoCo physics engine (Todorov *et al.*, 2012) and the framework provides an OpenAI Gym environment interface (Brockman *et al.*, 2016). Besides benchmarks, several baselines for meta- and multi-task RL algorithms are available for comparison. The provided analysis showed that most of the state-of-the-art approaches for RL struggle to learn more than a few tasks at the same time highlighting the difficulty of multi-task learning.

Robosuite Robosuite (Zhu *et al.*, 2020) is a simulation-based benchmark based on the MuJoCo physics engine but focused on robot learning. Its structure is modular, allowing for combining different robotic arms (e.g., Panda, two Sawyer arms), grippers, controllers, and sensors. It includes complex tasks such as bi-manual peg-in-hole, table wiping, and nut assembly; and it also counts with the addition of community-contributed tasks. Robosuite is used for imitation learning by Mandlekar *et al.* (2018), and in offline RL by Sinha *et al.* (2022).

RLBench RLBench (James *et al.*, 2020) is a learning benchmark for robot-arm manipulation tasks, featuring 100 hand-crafted tasks. It provides proprioceptive, RGB, and depth images for the learners. A CoppeliaSim/V-REP (Rohmer *et al.*, 2013) simulation environment is used for evaluating learners with a simulated Franka Emika's Panda robot arm. RLBench provides unique scalability through the use of motion planners to create an arbitrarily large number of synthetic demonstrations, a key enabling characteristic for RL and IL methods. RLBench is used to test and evaluate the interactive learning method by Chisari *et al.* (2022), who combine both corrective and evaluative feedback from the human teacher to asynchronously train a stochastic policy.

NIST Aiming to standardize the application which is used to test real-world robotic systems Kimble *et al.* (2020) propose the National Institute of Standards and Technology (NIST) benchmark, which is composed of a set of assembly tasks and their instructions. Instructions are provided to cheaply fabricate the parts of the robotic system, which are then used to reproduce the tasks. This benchmark has been used by Luo *et al.*, 2021, who propose an IIL method focused on industrial setups.

Conclusion The adoption of benchmarks for IIL has been limited, which is expected to be a direct consequence of the limitations in standardization due to the *human-in-the-loop* factor. Therefore, the adoption of the benchmarks presented in this section might require the design and implementation of human-centered evaluation metrics (see Section 10) in order pro provide a fair comparison between different methods. This section briefly covered benchmarks with promising applicability within the HIL scope for robotics. The interested reader is pointed to Stapelberg and Malan, 2020 for a survey on benchmarks for reinforcement learning, and to Zhang *et al.*, 2019a for a survey on benchmarks for deep learning.

11.4 Discussion

In this section, we highlight the main benchmarks and applications that have been approached through the lenses of IIL. The literature shows that IIL has a wide range of applicability, spanning from household, to medical, to industrial robots. We observe that many real-world applications require complex skills that are not easy to demonstrate, either due to the unavailability of a suitable interface or lack of demonstrator expertise. IIL methods show that they can provide a suitable framework to tackle these problems. However, it is still an open challenge to design methods that can handle imperfect feedback without issues.

Various benchmarks exist that can be used in order to evaluate and compare such algorithms in controlled settings. Nevertheless, due to the presence of the human teacher in the learning loop, exact and reproducible results are difficult to achieve, and still represent an open

challenge. Most benchmarks consist of simulated tasks, as they are the most portable and fastest mean to test new algorithms, ranging from arcade games to realistic robotic simulators. An exception is the NIST benchmark, which provides a standardized set of easily reproducible table-top tasks to be used for evaluating real-world robots.

12

Research Challenges and Opportunities

After discussing the different aspects involved in IIL, regarding ML algorithmic features, ways of interaction, interfaces, human factors, and evaluation considerations, we discuss some of the problems in the domain that represent a challenge and are potentially interesting directions for further research that could make the use of IIL for transferring the human knowledge to the machines more effectively.

- *Successfully combining multiple feedback modalities.* In Section 3, we presented the different modalities the teachers can use to convey their knowledge to the learning system and discussed the benefits and limitations of each of them, along with possible ways to choose which one is more convenient depending on the available teacher and the characteristics of the problem. However, there are situations wherein applying different kinds of feedback can be beneficial, e.g., using absolute corrective demonstrations when the user knows exactly what should be done, relative corrections for tuning the performed actions, and evaluative feedback for those states wherein the teacher is not sure what the right action is, but can still judge whether the agent is doing well or not. Providing those kinds of feedback seamlessly, with no specified schedule

within the same roll-out of the learning policy, is difficult to do with current methods, and almost all state-of-the-art methods focus on exploiting one kind of feedback. Furthermore, combining different datasets of different kinds of interactions, and combining smoothly and without conflicts different update rules depending on the information extracted from each kind of modality, remains an open challenge that has still a long path to follow.

- *Dealing with inconsistencies in the feedback.* Unlike other ML approaches, learning with humans in the loop has the specific problem of obtaining noisy data that is not only produced by observation and process noise, but also by the mistakes of the human teachers, which can be conflicting with data obtained in a different moment. Detecting feedback mistakes is a difficult problem that impacts safety and learning efficiency. For many methods, these mistakes condition the final policy performance. Some works have shown that different interaction modalities along with the learning schemes could be more robust to mistakes. Moreover, learning from MDP reward functions (RL) and human feedback have shown less sensitivity to teachers' mistakes. This problem, intrinsic to the human teachers, has been neglected by most state-of-the-art methods, which still work under the assumption of having perfect teachers, and have been evaluated with unrealistic perfect oracles. Efforts in this direction are required to obtain reliable learning agents for real situations and users.

- *Dealing with inconsistencies in the teaching strategy.* In IIL, the incremental learning process comes with the advantage that, through iterations, the teachers learn more about the problem at hand and the strategies to solve it. New knowledge from the teacher can result in feedback that is different with respect to the past for a specific situation. For instance, an action that used to be rewarded can later stop fitting the current policy; therefore, the teacher may consider punishing it, creating an inconsistency that is not produced by an occasional mistake. These kinds of situations are very likely to happen with real teachers. Nevertheless, it is challenging to evaluate learning methods regarding this kind

of inconsistency, because it is difficult to replicate a change of strategies with human teachers. Therefore, this issue represents an algorithmic challenge for detecting and solving the inconsistencies, but also a challenge from the evaluation procedure perspective. Some methods, indirectly deal with these inconsistencies, using a limited dataset of the most recent interactions that allow forgetting old data. However, this approach has the disadvantage of losing valuable and currently valid knowledge provided at the early stages of the learning process.

- *Dealing with inconsistencies from multimodal behaviors.* In some applications, like collaborative robots, the human operators could adapt the robot to their personal preferences. Then, the learning process collects data from different users who work with the robot but has to adapt to each of the users according to their preferences, while leveraging the knowledge contained in the data of the other users that is not conflicting with the current one's preference. The previous example has to do with problems that have multiple valid solutions, that can be demonstrated in different circumstances (even by the same teacher). In these cases, considerations should be taken in the model approximator and learning process, in order to capture the multimodality of the solution space. A more complex challenge results from being able to consider all the mentioned inconsistencies (in the feedback, the teaching strategy, and the multimodal solutions), and tackling each inconsistent data input with the correct strategy.

- *Safety during learning and policy deployment.* As in any HCI or HRI system, in IIL human safety is the main priority every time the system is used. The safety of the robot and the surrounding environment is a relevant consideration as well. In applications wherein the teaching process involves physical interactions, it is important to develop auxiliary models that support the safety of the system according to factors such as the current performance of the policy, the uncertainty of the policy, task-specific measures and interface restrictions.

- *Data efficiency.* This is a general problem in ML, especially when working with NNs, since training models based on NNs often requires a large amount of data, in particular for high-dimensional data. We have observed in the literature that, in general, IIL methods tend to be more data efficient than other strategies such as conventional IL or RL. However, with human users, this problem is more critical, since, to avoid demanding an unfeasible or unpleasant high user workload, the amount of data that can be obtained is limited. This is a general problem of IIL, and it could be indirectly approached when tackling some of the other problems listed in this section, or when employing auxiliary costs for training, pre-processing modules, pre-trained models, etc.

- *Teaching in high-dimensional spaces.* For human beings, handling many variables at the same time is a very difficult task in general (Halford *et al.*, 2005). In tasks wherein the agent has many degrees of freedom or many objects and conditions of the environment to consider, teachers could have a hard time processing what the exact actions or transitions to execute are. Therefore, learning strategies that help to reduce this complexity are desirable. Using evaluative feedback methods is a feasible solution in terms of the capabilities of the teachers because the feedback is reduced to one dimension. However, evaluative feedback does not guarantee a reduction of the solution space, and the required training time for a teacher to obtain a high-performance policy can be excessively long.

- *Realistic simulated teacher.* Evaluating and comparing IIL methods in realistic scenarios involves several human teachers participating in exhaustive experiments. In general, performing experiments with multiple participants interacting with the agents, using different learning methods, and repeating many learning processes, is unfeasible. In order to reduce the load on the participants, partial/preliminary evaluations can be based on experiments with simulated teachers that can be complemented with user studies. However, simulating all the human factors and the behaviors different kinds of people have in specific situations requires complex

human models that have not been studied in depth. Additionally, standardizing such realistic oracles would help to extrapolate the results and insights of previous works. Obtaining such results from experiments with perfect and unrealistic oracles bypasses the most difficult and challenging problem of IIL, which is learning from such a complex teacher.

- *Unified subjective measures.* IIL has been developed mostly by ML researchers; however, the domain also falls in the category of HCI or HRI. Research in IIL requires standard practices applied by the HCI or HRI communities; nevertheless, these considerations are not fully adopted in IIL. Although more and more works are including the analysis of some human factors with user studies, there is still a lack of standard protocols that agree on which the most convenient questionnaires, subjective metrics, and experimental setups are.

- *Unified benchmarks.* It is very common in IIL papers, that the evaluation environment is very specific to each work. There is a lack of common benchmarks to use in IIL research, as well as agreement on relevant problems for evaluating new methods. In the RL community, there is more progress in this direction, with a number of different simulated environments available as common test-beds. Those resources are useful for the IIL research community; however, they do not completely fulfill all the needs of this specific field of study, and more development is required in this direction.

13

Conclusion

In this work, we survey the most relevant works in the IIL literature, which have been developed for teaching robots or have potential benefits in robotic applications. In recent years, many works have shown the potential of these methods for enabling end-users with non-technical backgrounds to program or adapt the operation of robotic systems. The incremental component of these learning strategies has a positive impact on the usability of the methods and on the performance level of the obtained policies, with respect to traditional IL. The work provides a structure in the field that facilitates the comprehension of the concepts and problems involved. This organization can help to speed up the learning curve of the new researchers, but also improve the understanding and perspectives of established IIL practitioners.

Initially, we mention the problems regarding the ambiguous definitions in the terminology used by different authors and propose one unified terminology, along with the resulting classification of learning schemes that allows specifying what methods that learn from teachers can be considered IIL. We group the algorithms according to different important aspects such as the information provided by the teacher during the interaction, the information extracted and modeled during

the interaction by the learning agent, the way data obtained is handled for learning, the interfaces used for communicating with teachers and learners, the relation IIL can have with RL, the model representations used for abstracting the obtained knowledge, the considerations that should be taken when having humans in the loop and the ways to evaluate their experience, and few other considerations. Later on, we present most of the benchmarks that can be used in the experimental design of IIL methods and the most relevant applications that have been tackled with this kind of strategy. All these aspects have to be considered when selecting, designing, implementing, or testing a method. Finally, we discuss some of the challenges that researchers in this field of study still need to address in order to directly or indirectly improve the learning performance of the agent and the teachers' experience, which can be regarded as the general objectives that the research community aims to achieve.

Author Contributions

Carlos Celemin designed the structure of this work, worked on Sections 1.1, 1.2, 3.1.1, 3.1.2, 3.2.1, 3.2.2, 3.3, 7.1, 7.4, Sections 12, 13, sketching the figures, the internal review process, and the final rewrite of the document.
Rodrigo Pérez-Dattari worked on Sections 2.2, 4.1, 4.4, 6.5, 7.2, 7.3, the data management, the internal review process, and the final rewrite of the document.
Eugenio Chisari worked on Sections 3.1.1, 3.1.2, 3.2.1, 4.3, 5.3, 5.5, the internal review process, and the final rewrite of the document.
Giovanni Franzese worked on Sections 1.3, 4.2, 4.3, 5.3, 5.4, Section 6, sketching the figures, and the internal review process.
Leandro de Souza Rosa worked on Sections 5.1, 11.2, 11.3, Section 9, the data management, and the internal review process.
Ravi Prakash worked on Sections 5.2, 11.1, Section 10, and the internal review process.
Zlatan Ajanović worked on Sections 2.1, 5.2, 11.3, Section 8, sketching the figures, and the internal review process.
Marta Ferraz worked on Section 9.3, Section 10, and the internal review process.
Abhinav Valada supervised the development of this project and contributed to the internal review process.
Jens Kober supervised the development of this project and contributed to the internal review process.

Acknowledgments

This research has been funded by the Netherlands Organization for Scientific Research (NWO) project FlexCRAFT, grant number P17-01, by the ERC Stg TERI, project reference #804907, as well as by the BrainLinks-BrainTools center of the University of Freiburg.

Glossary

A-OPI Advice-Operator Policy Improvement. 48

ACTAMER Actor-Critic TAMER. 32

AEUS Expected Utility Selection. 63

AggreVaTe Aggregate Values To Imitate. 107

AI Artificial Intelligence. 10, 101

ASR Automatic Speech Recognition. 115

BC Behavioral Cloning. 10–13, 21, 22, 24, 40, 135

BCO Behavioral Cloning from Observations. 49

CEILing Corrective and Evaluative Interactive Learning. 43, 116

COACHc COrrective Advice Communicated by Humans. 33, 48, 49, 96, 97, 114

COACHe Convergent Actor-Critic by Humans. 33, 48, 95, 99

D-COACH Deep COACH. 49, 67, 68, 139

D-TAMER Deep TAMER. 32, 67

DAgger Data Aggregation. 25, 41–43, 45, 58, 74, 81, 82, 94, 96, 97, 107, 112

DDPGfD Deep Deterministic Policy Gradient from Demonstration. 44

DemPref Learning Reward Functions by Integrating Human Demonstrations and Preferences. 38, 62

DMP Dynamic Movement Primitive. 60, 61, 83, 84

DoF Degree of Freedom. 60, 70, 112, 113, 133, 135–137

DP Dynamic Programming. 91

DPL Direct Policy Learning. 57–59

DQfD Deep Q-learning from demonstrations. 101

DSTL Desired State Transition Learning. 59, 60

EfD Exploration from Demonstration. 105

EIL Expert Intervention Learning. 44

EnsembleDAgger Ensemble Dagger. 45, 46

FERL Feature Expansive Reward Learning. 68, 111

FRESH Feedback-based REward SHaping. 107

GA Genetic Algorithm. 29, 30

GMM Gaussian Mixture Model. 81, 84

GP Gaussian Process. 60, 80, 81, 84, 85

GUI Graphical User Interface. 105

HCAI Human Centered Artificial Intelligence. 10

HCI Human-Computer Interaction. 10, 153, 155

HCML Human Centered Machine Learning. 10, 127

HG-DAgger Human Gated DAgger. 43, 44, 58

HIL Human in the Loop. 131, 146, 149

HIL-AI Human in the Loop Artificial Intelligence. 10

HIL-ML Human in the Loop Machine Learning. 9

HIL-RL Human in the Loop Reinforcement Learning. 12

HRI Human-Robot Interaction. 10, 50, 153, 155

I-SABL Inferring Strategy-Aware Bayesian Learning. 33, 75

IAI Interactive Artificial Intelligence. 10

IEC Interactive Evolutionary Computation. 29, 30

IER Interactive Evolutionary Robotics. 30

IIL Interactive Imitation Learning. 4–8, 11–16, 19–27, 51, 55–57, 59–61, 64, 65, 67, 68, 70–72, 76, 77, 79, 82–85, 87, 88, 93–97, 99, 101, 108, 110–122, 124, 127, 128, 130–137, 139, 141, 143–147, 149–157

IL Imitation Learning. 3, 5–13, 23–25, 51, 86–89, 93, 99, 107, 135, 149, 154, 156

ILS Interactive Learning Systems. 11, 12

IML Interactive Machine Learning. 8–11, 118, 120

Interactive RL Interactive Reinforcement Learning. 7, 31, 106

IRL Inverse Reinforcement Learning. 10, 11, 13, 62, 68, 73, 101

IWR Intervention Weighted Regression. 43

KL Kullback–Leibler. 96

KMP Kernelized Movement Primitive. 84

LaND Learning to Navigate from Disengagements. 44, 62, 72

LazyDAgger Lazy DAgger. 45, 74

LBW Learning By Watching. 102

LEC Learning with an External Critic. 102

LfC Learning from Critique. 7

LfD Learning from Demonstration. 7, 8, 29, 40, 114, 148

LfP Learning from Preferences. 112, 117

LIRA Learning Interactively to Resolve Ambiguity. 111

LOKI Locally Optimal search after K-step Imitation. 101

LSTM Long Short-Term Memory. 68

LWR Locally Weighed Regression. 79, 80

MAP Maximum a Posteriori. 62

MC Monte Carlo. 91

MDP Markov Decision Process. 16–18, 20, 30, 32, 61, 86, 98, 152

ML Machine Learning. 3, 5, 7–11, 54, 71, 98, 118, 122, 127, 130, 151, 152, 154, 155

MLE Maximum Likelihood Estimation. 24, 25, 96

MP Movement Primitive. 84

MPC Model Predictive Control. 8

MSE Mean Squared Error. 25

NIST National Institute of Standards and Technology. 149, 150

NN Neural Network. 36, 49, 60, 68, 73, 81–83, 85, 98, 107, 148, 154

PbD Programming by Demonstrations or Programming by Doing. 7, 8

POMDP Partially Observable Markov Decision Process. 18, 79

PoWER Policy Learning by Weighting Exploration with the Returns. 101

PPL Preference-Based Policy Learning. 35, 63

ProMP Probabilistic Movement Primitive. 60, 84, 85

PS Policy Search. 83, 105, 113, 135, 136

RBF Radial Basis Function. 79

RL Reinforcement Learning. 3, 10–13, 20, 23, 24, 29–32, 36, 48, 49, 59, 61, 82, 86–89, 91, 93–95, 97, 98, 100–102, 104–108, 113, 116, 136, 147–149, 152, 154, 155, 157

RL-HiL RL with Human-in-the-Loop. 29, 101–103, 108

SABL Strategy-Aware Bayesian Learning. 33

Safe-RL Safe Reinforcement Learning. 106

SafeDAgger Safe DAgger. 45, 46, 74

SHIELD Super-Human InsErtion using Learning from Demonstration. 44

SHIV Svm-based reduction in Human InterVention. 42

SRL State Representation Learning. 83

SVM Support Vector Machine. 81

TAMER Training an Agent Manually via Evaluative Reinforcement. 32, 63, 74, 95, 96, 98, 107, 112, 114, 139

TD Temporal-Difference. 91, 92

TD-DIS Temporal-Difference per Decision Importance Sampling. 92, 93

ThriftyDAgger Thrifty DAgger. 46

TICS Task-Instruction-Contingency-Shaping. 114

TIPS Teaching Imitative Policies in State-space. 49, 72

TPC Tactile Policy Correction. 50

TPP Trajectory Preference Perceptron. 50, 62, 71

VR Virtual Reality. 117, 147

References

Abdel-Malek, K., J. Yang, W. Yu, and J. Duncan. (2005). "Human performance measures: mathematics". *Department of Mechanical Engineering The University of Iowa, Technical report*: 1–27.

Ablett, T., F. Marić, and J. Kelly. (2020). "Fighting Failures with FIRE: Failure Identification to Reduce Expert Burden in Intervention-Based Learning". *arXiv preprint arXiv:2007.00245*.

Akrour, R., M. Schoenauer, and M. Sebag. (2012). "APRIL: Active Preference Learning-Based Reinforcement Learning". In: *Machine Learning and Knowledge Discovery in Databases*. Berlin, Heidelberg: Springer Berlin Heidelberg. 116–131.

Akrour, R., M. Schoenauer, and M. Sebag. (2011). "Preference-Based Policy Learning". In: *Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part I. ECML PKDD'11*. Athens, Greece: Springer-Verlag. 12–27.

Akrour, R., M. Schoenauer, M. Sebag, and J.-C. Souplet. (2014). "Programming by Feedback". In: *International Conference on Machine Learning. JMLR Proceedings*. No. 32. Pékin, China: JMLR.org. 1503–1511. URL: https://hal.inria.fr/hal-00980839.

Alshiekh, M., R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu. (2018). "Safe Reinforcement Learning via Shielding". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. No. 1.

Aly, A., S. Griffiths, and F. Stramandinoli. (2017). "Metrics and benchmarks in human-robot interaction: Recent advances in cognitive robotics". *Cognitive Systems Research*. 43: 313–323. DOI: https://doi.org/10.1016/j.cogsys.2016.06.002.

Amershi, S., M. Cakmak, W. B. Knox, and T. Kulesza. (2014). "Power to the people: The role of humans in interactive machine learning". *AI Magazine*. 35(4): 105–120.

Amershi, S., J. Fogarty, and D. Weld. (2012). "Regroup: Interactive machine learning for on-demand group creation in social networks". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 21–30.

Amodei, D., C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. (2016). "Concrete Problems in AI Safety". DOI: 10.48550/arXiv.1606.06565.

Arakawa, R., S. Kobayashi, Y. Unno, Y. Tsuboi, and S.-i. Maeda. (2018). "DQN-TAMER: Human-in-the-Loop Reinforcement Learning with Intractable Feedback". DOI: 10.48550/ARXIV.1810.11748.

Argall, B. D. (2009). "Learning mobile robot motion control from demonstration and corrective feedback". *PhD thesis*. Carnegie Mellon University.

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

Argall, B. D., B. Browning, and M. Veloso. (2008). "Learning robot motion control with demonstration and advice-operators". In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 399–404. DOI: 10.1109/IROS.2008.4651020.

Argall, B. D., B. Browning, and M. M. Veloso. (2011a). "Teacher feedback to scaffold and refine demonstrated motion primitives on a mobile robot". *Robotics and Autonomous Systems*. 59(3): 243–255. DOI: https://doi.org/10.1016/j.robot.2010.11.004.

Argall, B. D., E. L. Sauser, and A. G. Billard. (2011b). "Tactile Guidance for Policy Adaptation". *Foundations and Trends® in Robotics*. 1(2): 79–133. DOI: 10.1561/2300000012.

Arumugam, D., J. K. Lee, S. Saskin, and M. L. Littman. (2019). "Deep Reinforcement Learning from Policy-Dependent Human Feedback". DOI: 10.48550/ARXIV.1902.04257.

Arzate Cruz, C. and T. Igarashi. (2020). "A Survey on Interactive Reinforcement Learning: Design Principles and Open Challenges". In: Proceedings of the 2020 ACM Designing Interactive Systems Conference. 1195–1209.

Bain, M. and C. Sammut. (1995). "A Framework for Behavioural Cloning." In: *Machine Intelligence 15*. 103–129.

Bajcsy, A., D. P. Losey, M. K. O'Malley, and A. D. Dragan. (2018). "Learning from physical human corrections, one feature at a time". In: *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. 141–149.

Bajcsy, A., D. P. Losey, M. K. O'Malley, and A. D. Dragan. (2017). "Learning robot objectives from physical human interaction". In: *Conference on Robot Learning*. PMLR. 217–226.

Balakrishna, A., B. Thananjeyan, J. Lee, F. Li, A. Zahed, J. E. Gonzalez, and K. Goldberg. (2020). "On-policy robot imitation learning from a converging supervisor". In: *Conference on Robot Learning*. PMLR. 24–41.

Beel, J., B. Gipp, S. Langer, and C. Breitinger. (2016). "Paper recommender systems: a literature survey". *International Journal on Digital Libraries*. 17(4): 305–338.

Behnke, S. (2006). "Robot competitions-ideal benchmarks for robotics research". In: *Proc. of IROS-2006 Workshop on Benchmarks in Robotics Research*. Institute of Electrical and Electronics Engineers (IEEE).

Bellman, R. (1957). "A Markovian decision process". *Journal of Mathematics and Mechanics*: 679–684.

Ben Amor, H., E. Berger, D. Vogt, and B. Jung. (2009). "Kinesthetic Bootstrapping: Teaching Motor Skills to Humanoid Robots through Physical Interaction". In: *KI 2009: Advances in Artificial Intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg. 492–499.

Billard, A., S. Calinon, R. Dillmann, and S. Schaal. (2008). "Survey: Robot programming by demonstration". *Handbook of robotics*. 59(BOOK_CHAP).

Billard, A. and D. Grollman. (2013). "Robot learning by demonstration". *Scholarpedia*. 8(12): 3824.

Billard, A. G., S. Calinon, and R. Dillmann. (2016). "Learning from humans". In: *Springer handbook of robotics*. Springer. 1995–2014.

Billing, E. A. and T. Hellström. (2010). "A formalism for learning from demonstration". *Paladyn, Journal of Behavioral Robotics*. 1(1): 1–13.

Bishop, C. M. (2006). "Pattern recognition". *Machine learning*. 128(9).

Biyik, E., M. Palan, N. C. Landolfi, D. P. Losey, and D. Sadigh. (2020). "Asking Easy Questions: A User-Friendly Approach to Active Reward Learning". In: *Proceedings of the Conference on Robot Learning*. Vol. 100. *Proceedings of Machine Learning Research*. PMLR. 1177–1190. URL: https://proceedings.mlr.press/v100/b-iy-ik20a.html.

Biyik, E. and D. Sadigh. (2018). "Batch Active Preference-Based Learning of Reward Functions". In: *Proceedings of The 2nd Conference on Robot Learning*. Vol. 87. *Proceedings of Machine Learning Research*. PMLR. 519–528. URL: https://proceedings.mlr.press/v87/biyik18a.html.

Bıyık, E., N. Huynh, M. J. Kochenderfer, and D. Sadigh. (2020). "Active Preference-Based Gaussian Process Regression for Reward Learning". DOI: 10.48550/ARXIV.2005.02575.

Blukis, V., N. Brukhim, A. Bennett, R. A. Knepper, and Y. Artzi. (2018). "Following high-level navigation instructions on a simulated quadcopter with imitation learning". *arXiv preprint arXiv:1806.00047*.

Blumberg, B., M. Downie, Y. Ivanov, M. Berlin, M. P. Johnson, and B. Tomlinson. (2002). "Integrated Learning for Interactive Synthetic Characters". In: *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '02*. San Antonio, Texas: Association for Computing Machinery. 417–426. DOI: 10.1145/566570.566597.

Bobadilla, J., F. Ortega, A. Hernando, and A. Gutiérrez. (2013). "Recommender systems survey". *Knowledge-based systems*. 46: 109–132.

Bobu, A. and A. Peng. (2022). "Aligning Robot Representations with Humans". DOI: 10.48550/ARXIV.2205.07882.

Bobu, A., M. Wiggert, C. Tomlin, and A. D. Dragan. (2021). "Feature Expansive Reward Learning: Rethinking Human Input". In: *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction. HRI '21*. Boulder, CO, USA: Association for Computing Machinery. 216–224. DOI: 10.1145/3434073.3444667.

Bobu, A., M. Wiggert, C. Tomlin, and A. D. Dragan. (2022). "Inducing Structure in Reward Learning by Learning Features". *The International Journal of Robotics Research*. 0(0): 02783649221078031. DOI: 10.1177/02783649221078031.

Böhmer, W., J. T. Springenberg, J. Boedecker, M. Riedmiller, and K. Obermayer. (2015). "Autonomous learning of state representations for control: An emerging field aims to autonomously learn state representations for reinforcement learning agents from their real-world sensor observations". *KI-Künstliche Intelligenz*. 29(4): 353–362.

Bootsma, B., G. Franzese, and J. Kober. (2021). "Interactive learning of sensor policy fusion". In: *2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*. IEEE. 665–670.

Bouthillier, X., P. Delaunay, M. Bronzi, A. Trofimov, B. Nichyporuk, J. Szeto, N. Mohammadi Sepahvand, E. Raff, K. Madan, V. Voleti, S. Ebrahimi Kahou, V. Michalski, T. Arbel, C. Pal, G. Varoquaux, and P. Vincent. (2021). "Accounting for Variance in Machine Learning Benchmarks". In: *Proceedings of Machine Learning and Systems*. Vol. 3. 747–769. URL: https://proceedings.mlsys.org/paper/2021/file/cfecdb276f634854f3ef915e2e980c31-Paper.pdf.

Brockman, G., V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. (2016). "OpenAI Gym". DOI: 10.48550/ARXIV.1606.01540.

Brown, D. S., Y. Cui, and S. Niekum. (2018). "Risk-aware active inverse reinforcement learning". In: *Conference on Robot Learning*. PMLR. 362–372.

Brown, D. S., R. Coleman, R. Srinivasan, and S. Niekum. (2020). "Safe Imitation Learning via Fast Bayesian Reward Inference from Preferences". In: *Proceedings of the 37th International Conference on Machine Learning. ICML'20*. JMLR.org.

Burke, R. (2002). "Hybrid recommender systems: Survey and experiments". *User modeling and user-adapted interaction*. 12(4): 331–370.

Cakmak, M. and A. L. Thomaz. (2012). "Designing robot learners that ask good questions". In: *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*. ACM. 17–24.

Calinon, S. (2018). "Learning from demonstration (programming by demonstration)". *Encyclopedia of Robotics*: 1–8.

Canal, G., G. Alenyà, and C. Torras. (2016). "Personalization Framework for Adaptive Robotic Feeding Assistance". In: *Social Robotics*. Cham: Springer International Publishing. 22–31.

Canal, G., E. Pignat, G. Alenyà, S. Calinon, and C. Torras. (2018). "Joining high-level symbolic planning with low-level motion primitives in adaptive HRI: application to dressing assistance". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 3273–3278. DOI: 10.1109/ICRA.2018.8460606.

Canal, G., C. Torras, and G. Alenyà. (2021). "Are Preferences Useful for Better Assistance?: A Physically Assistive Robotics User Study". *ACM Transactions on Human-Robot Interaction (THRI)*. 10(4): 1–19. DOI: 10.1145/3472208.

Cederborg, T., I. Grover, C. L. Isbell Jr, and A. L. Thomaz. (2015). "Policy Shaping with Human Teachers." In: *IJCAI*. 3366–3372.

Celemin, C., G. Maeda, J. Ruiz-del-Solar, J. Peters, and J. Kober. (2019a). "Reinforcement learning of motor skills using policy search and human corrective advice". *The International Journal of Robotics Research*. 38(14): 1560–1580.

Celemin, C. and J. Ruiz-del-Solar. (2015). "COACH: Learning continuous actions from COrrective Advice Communicated by Humans". In: *2015 International Conference on Advanced Robotics (ICAR)*. 581–586. DOI: 10.1109/ICAR.2015.7251514.

Celemin, C. and J. Ruiz-del-Solar. (2019). "An interactive framework for learning continuous actions policies based on corrective feedback". *Journal of Intelligent & Robotic Systems*. 95(1): 77–97. DOI: 10.1007/s10846-018-0839-z.

Celemin, C., J. Ruiz-del-Solar, and J. Kober. (2019b). "A fast hybrid reinforcement learning framework with human corrective feedback". *Autonomous Robots*. 43(5): 1173–1186.

Chang, K.-W., A. Krishnamurthy, A. Agarwal, H. Daumé III, and J. Langford. (2015). "Learning to search better than your teacher". In: *International Conference on Machine Learning*. PMLR. 2058–2066.

Chatzimparmpas, A., R. M. Martins, I. Jusufi, and A. Kerren. (2020). "A survey of surveys on the use of visualization for interpreting machine learning models". *Information Visualization*. 19(3): 207–233.

Cheng, C.-A., X. Yan, N. Wagener, and B. Boots. (2018). "Fast Policy Learning through Imitation and Reinforcement". DOI: 10.48550/ARXIV.1805.10413.

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

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

Chisari, E., T. Welschehold, J. Boedecker, W. Burgard, and A. Valada. (2022). "Correct me if i am wrong: Interactive learning for robotic manipulation". *IEEE Robotics and Automation Letters*. 7(2): 3695–3702.

Christiano, P. F., J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. (2017). "Deep Reinforcement Learning from Human Preferences". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.

Chu, V., B. Akgun, and A. L. Thomaz. (2016). "Learning haptic affordances from demonstration and human-guided exploration". In: *2016 IEEE Haptics Symposium (HAPTICS)*. 119–125. DOI: 10.1109/HAPTICS.2016.7463165.

Chu, V. and A. L. Thomaz. (2015). "Exploring Affordances Using Human-Guidance and Self-Exploration". In: *2015 AAAI Fall Symposium Series*.

Clouse, J. A. and P. E. Utgoff. (1992). "A Teaching Method for Reinforcement Learning". In: *Machine Learning Proceedings 1992*. Elsevier. 92–101.

Cohn, D. A., Z. Ghahramani, and M. I. Jordan. (1996). "Active learning with statistical models". *Journal of artificial intelligence research*. 4: 129–145.

Corrigan, L. J., C. Peters, D. Küster, and G. Castellano. (2016). "Engagement Perception and Generation for Social Robots and Virtual Agents". In: *Toward Robotic Socially Believable Behaving Systems - Volume I : Modeling Emotions*. Cham: Springer International Publishing. 29–51. DOI: 10.1007/978-3-319-31056-5_4.

Coumans, E. (2015). "Bullet Physics Simulation". In: *ACM SIGGRAPH 2015 Courses. SIGGRAPH '15*. Los Angeles, California: Association for Computing Machinery. DOI: 10.1145/2776880.2792704.

Cronrath, C., E. Jorge, J. Moberg, M. Jirstrand, and B. Lennartson. (2018). "BAgger: A Bayesian algorithm for safe and query-efficient imitation learning". In: *Machine Learning in Robot Motion Planning–IROS 2018 Workshop*.

Cruz, F., G. I. Parisi, and S. Wermter. (2018). "Multi-modal Feedback for Affordance-driven Interactive Reinforcement Learning". In: *2018 International Joint Conference on Neural Networks (IJCNN)*. 1–8. DOI: 10.1109/IJCNN.2018.8489237.

Cruz, F., J. Twiefel, S. Magg, C. Weber, and S. Wermter. (2015). "Interactive reinforcement learning through speech guidance in a domestic scenario". In: *2015 International Joint Conference on Neural Networks (IJCNN)*. 1–8. DOI: 10.1109/IJCNN.2015.7280477.

Cuayáhuitl, H., M. van Otterlo, N. Dethlefs, and L. Frommberger. (2013). "Machine learning for interactive systems and robots: a brief introduction". In: *Proceedings of the 2nd Workshop on Machine Learning for Interactive Systems: Bridging the Gap Between Perception, Action and Communication*. ACM. 19–28.

Cui, Y., D. Isele, S. Niekum, and K. Fujimura. (2019). "Uncertainty-aware data aggregation for deep imitation learning". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 761–767.

Cui, Y., P. Koppol, H. Admoni, S. Niekum, R. Simmons, A. Steinfeld, and T. Fitzgerald. (2021). "Understanding the Relationship between Interactions and Outcomes in Human-in-the-Loop Machine Learning". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, Montreal, QC, Canada*. Vol. 10.

Cui, Y. and S. Niekum. (2018). "Active Reward Learning from Critiques". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 6907–6914. DOI: 10.1109/ICRA.2018.8460854.

Curnow, H. J. and B. A. Wichmann. (1976). "A synthetic benchmark". *The Computer Journal*. 19(1): 43–49. DOI: 10.1093/comjnl/19.1.43.

CWI, I. and G. Amsterdam. (1997). "Cellular encoding for interactive evolutionary robotics". In: *Fourth European conference on artificial life*. Vol. 4. MIT Press. 368.

Daniel, C., O. Kroemer, M. Viering, J. Metz, and J. Peters. (2015). "Active Reward Learning with a Novel Acquisition Function". *Autonomous Robots*. 39(3): 389–405. DOI: 10.1007/s10514-015-9454-z.

Dasari, S., F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. (2020). "RoboNet: Large-Scale Multi-Robot Learning". In: *Proceedings of the Conference on Robot Learning*. Vol. 100. *Proceedings of Machine Learning Research*. PMLR. 885–897. URL: https://proceedings.mlr.press/v100/dasari20a.html.

Degris, T., M. White, and R. Sutton. (2012). "Off-Policy Actor-Critic". In: *International Conference on Machine Learning*.

Deisenroth, M. P., G. Neumann, J. Peters, *et al.* (2013). "A survey on policy search for robotics". *Foundations and Trends® in Robotics*. 2(1–2): 1–142. URL: http://dx.doi.org/10.1561/2300000021.

Della Santina, C., C. Piazza, G. Grioli, M. G. Catalano, and A. Bicchi. (2018). "Toward dexterous manipulation with augmented adaptive synergies: The pisa/iit softhand 2". *IEEE Transactions on Robotics*. 34(5): 1141–1156.

DelPreto, J., J. I. Lipton, L. Sanneman, A. J. Fay, C. Fourie, C. Choi, and D. Rus. (2020). "Helping robots learn: a human-robot master-apprentice model using demonstrations via virtual reality teleoperation". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 10226–10233.

Dosovitskiy, A., G. Ros, F. Codevilla, A. Lopez, and V. Koltun. (2017). "CARLA: An open urban driving simulator". In: *Conference on robot learning*. PMLR. 1–16.

Dudley, J. J. and P. O. Kristensson. (2018). "A Review of User Interface Design for Interactive Machine Learning". *ACM Trans. Interact. Intell. Syst.* 8(2). DOI: 10.1145/3185517.

Ewerton, M., G. Maeda, G. Kollegger, J. Wiemeyer, and J. Peters. (2016). "Incremental imitation learning of context-dependent motor skills". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. 351–358. DOI: 10.1109/HUMANOIDS.2016.7803300.

Fails, J. A. and D. R. Olsen Jr. (2003). "Interactive machine learning". In: *Proceedings of the 8th international conference on Intelligent user interfaces*. ACM. 39–45.

Fan, L., Y. Zhu, J. Zhu, Z. Liu, O. Zeng, A. Gupta, J. Creus-Costa, S. Savarese, and L. Fei-Fei. (2018). "SURREAL: Open-Source Reinforcement Learning Framework and Robot Manipulation Benchmark". In: *Proceedings of The 2nd Conference on Robot Learning*. Vol. 87. *Proceedings of Machine Learning Research*. PMLR. 767–782. URL: https://proceedings.mlr.press/v87/fan18a.html.

Ferraz, M., E. Ferreira, E. d. Exter, F. v. d. Hulst, H. Rovina, W. Carey, J. Grenouilleau, and T. Krueger. (2019). "Multisensory real-time space telerobotics". In: *Intelligent Computing-Proceedings of the Computing Conference*. Springer. 275–298.

Finn, C., S. Levine, and P. Abbeel. (2016). "Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48. ICML'16*. New York, NY, USA: JMLR.org. 49–58.

Fitzgerald, T., K. Bullard, A. Thomaz, and A. Goel. (2016). "Situated mapping for transfer learning". In: *Fourth Annual Conference on Advances in Cognitive Systems*.

Fitzgerald, T., A. Goel, and A. Thomaz. (2018). "Human-Guided Object Mapping for Task Transfer". *J. Hum.-Robot Interact.* 7(2). DOI: 10.1145/3277905.

Fleming, P. J. and J. J. Wallace. (1986). "How Not to Lie with Statistics: The Correct Way to Summarize Benchmark Results". *Commun. ACM*. 29(3): 218–221. DOI: 10.1145/5666.5673.

Franzese, G., C. Celemin, and J. Kober. (2021a). "Learning Interactively to Resolve Ambiguity in Reference Frame Selection". In: *Proceedings of the 2020 Conference on Robot Learning*. Vol. 155. *Proceedings of Machine Learning Research*. PMLR. 1298–1311. URL: https://proceedings.mlr.press/v155/franzese21a.html.

Franzese, G., A. Mészáros, L. Peternel, and J. Kober. (2021b). "ILoSA: Interactive Learning of Stiffness and Attractors": 7778–7785. DOI: 10.1109/IROS51168.2021.9636710.

Fürnkranz, J., E. Hüllermeier, W. Cheng, and S.-H. Park. (2012). "Preference-based reinforcement learning: a formal framework and a policy iteration algorithm". *Machine learning*. 89(1-2): 123–156.

Garcıa, J. and F. Fernández. (2015). "A Comprehensive Survey on Safe Reinforcement Learning". *Journal of Machine Learning Research*. 16(1): 1437–1480.

Ghasemipour, S. K. S., R. Zemel, and S. Gu. (2020). "A divergence minimization perspective on imitation learning methods". In: *Conference on Robot Learning*. PMLR. 1259–1277.

Gibson, J. J. (1977). "The theory of affordances". *Hilldale, USA*. 1(2): 67–82.

Goecks, V. G., G. M. Gremillion, V. J. Lawhern, J. Valasek, and N. R. Waytowich. (2019). "Efficiently Combining Human Demonstrations and Interventions for Safe Training of Autonomous Systems in Real-Time". In: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence. AAAI'19/IAAI'19/EAAI'19*. Honolulu, Hawaii, USA: AAAI Press. DOI: 10.1609/aaai.v33i01.33012462.

Goodfellow, I., Y. Bengio, and A. Courville. (2016). *Deep learning*. MIT press.

Griffith, S., K. Subramanian, J. Scholz, C. L. Isbell, and A. Thomaz. (2013). "Policy Shaping: Integrating Human Feedback with Reinforcement Learning". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2. NIPS'13*. Lake Tahoe, Nevada: Curran Associates Inc. 2625–2633.

Grizou, J., M. Lopes, and P.-Y. Oudeyer. (2013). "Robot learning simultaneously a task and how to interpret human instructions". In: *2013 IEEE Third Joint International Conference on Development and Learning and Epigenetic Robotics (ICDL)*. 1–8. DOI: 10.1109/DevLrn.2013.6652523.

Haario, H., E. Saksman, and J. Tamminen. (2001). "An adaptive Metropolis algorithm". *Bernoulli*: 223–242.

Halford, G. S., R. Baker, J. E. McCredden, and J. D. Bain. (2005). "How many variables can humans process?" *Psychological science*. 16(1): 70–76.

Hammersley, J. and D. Handscomb. (1964). "Monte carlo methods, methuen & co". *Ltd., London*. 40.

Haykin, S. S. (2001). *Neural networks: a comprehensive foundation*. Tsinghua University Press.

He, X., H. Chen, and B. An. (2020). "Learning Behaviors with Uncertain Human Feedback". In: *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*. Vol. 124. *Proceedings of Machine Learning Research*. PMLR. 131–140. URL: https://proceedings.mlr.press/v124/he20a.html.

Hedlund, E., M. Johnson, and M. Gombolay. (2021). "The Effects of a Robot's Performance on Human Teachers for Learning from Demonstration Tasks". In: *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction. HRI '21*. Boulder, CO, USA: Association for Computing Machinery. 207–215. DOI: 10.1145/3434073.3444664.

Hester, T., M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, *et al.* (2018). "Deep q-learning from demonstrations". In: *Thirty-second AAAI conference on artificial intelligence*.

Hindle, B. R., J. W. Keogh, and A. V. Lorimer. (2021). "Inertial-based human motion capture: A technical summary of current processing methodologies for spatiotemporal and kinematic measures". *Applied Bionics and Biomechanics*. 2021.

Ho, M. K., M. L. Littman, F. Cushman, and J. L. Austerweil. (2015). "Teaching with rewards and punishments: Reinforcement or communication?" In: *CogSci*.

Holzinger, A. (2016). "Interactive machine learning for health informatics: when do we need the human-in-the-loop?" *Brain Informatics*. 3(2): 119–131.

Hoque, R., A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg. (2022). "ThriftyDAgger: Budget-Aware Novelty and Risk Gating for Interactive Imitation Learning". In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. *Proceedings of Machine Learning Research*. PMLR. 598–608. URL: https://proceedings.mlr.press/v164/hoque22a.html.

Hoque, R., A. Balakrishna, C. Putterman, M. Luo, D. S. Brown, D. Seita, B. Thananjeyan, E. Novoseller, and K. Goldberg. (2021). "LazyDAgger: Reducing Context Switching in Interactive Imitation Learning". In: *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. 502–509. DOI: 10.1109/CASE49439.2021.9551469.

Howard, R. A. (1960). "Dynamic programming and markov processes".

Huang, Y., L. Rozo, J. Silvério, and D. G. Caldwell. (2019). "Kernelized movement primitives". *The International Journal of Robotics Research*. 38(7): 833–852.

Hurtado, J. V., L. Londoño, and A. Valada. (2021). "From Learning to Relearning: A Framework for Diminishing Bias in Social Robot Navigation". *Frontiers in Robotics and AI*. 8. DOI: 10.3389/frobt.2021.650325.

Hussein, A., M. M. Gaber, E. Elyan, and C. Jayne. (2017). "Imitation learning: A survey of learning methods". *ACM Computing Surveys (CSUR)*. 50(2): 1–35.

Ibarz, B., J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei. (2018). "Reward Learning from Human Preferences and Demonstrations in Atari". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems. NIPS'18*. Montréal, Canada: Curran Associates Inc. 8022–8034.

Isbell, C. and C. Shelton. (2001). "Cobot: A Social Reinforcement Learning Agent". In: *Advances in Neural Information Processing Systems*. Vol. 14. MIT Press. URL: https://proceedings.neurips.cc/paper/2001/hash/92bbd31f8e0e43a7da8a6295b251725f-Abstract.html (accessed on 04/05/2022).

Jain, A., S. Sharma, T. Joachims, and A. Saxena. (2015). "Learning preferences for manipulation tasks from online coactive feedback". *The International Journal of Robotics Research*. 34(10): 1296–1313.

Jain, A., B. Wojcik, T. Joachims, and A. Saxena. (2013). "Learning Trajectory Preferences for Manipulators via Iterative Improvement". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1. NIPS'13*. Lake Tahoe, Nevada: Curran Associates Inc. 575–583.

James, S., Z. Ma, D. R. Arrojo, and A. J. Davison. (2020). "RLBench: The Robot Learning Benchmark amp; Learning Environment". *IEEE Robotics and Automation Letters*. 5(2): 3019–3026. DOI: 10.1109/LRA.2020.2974707.

Jang, E., A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. (2022). "BC-Z: Zero-Shot Task Generalization with Robotic Imitation Learning". In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. *Proceedings of Machine Learning Research*. PMLR. 991–1002. URL: https://proceedings.mlr.press/v164/jang22a.html.

Jauhri, S., C. Celemin, and J. Kober. (2021). "Interactive Imitation Learning in State-Space". In: *Proceedings of the 2020 Conference on Robot Learning*. Vol. 155. *Proceedings of Machine Learning Research*. PMLR. 682–692. URL: https://proceedings.mlr.press/v155/jauhri21a.html.

Jiang, L., S. Liu, and C. Chen. (2019). "Recent research advances on interactive machine learning". *Journal of Visualization*. 22(2): 401–417.

Kaelbling, L. P., M. L. Littman, and A. R. Cassandra. (1998). "Planning and acting in partially observable stochastic domains". *Artificial Intelligence*. 101(1): 99–134. DOI: 10.1016/S0004-3702(98)00023-X.

Kahn, G., P. Abbeel, and S. Levine. (2021). "LaND: Learning to Navigate From Disengagements". *IEEE Robotics and Automation Letters*. 6(2): 1872–1879. DOI: 10.1109/LRA.2021.3060404.

Kamohara, S., H. Takagi, and T. Takeda. (1997). "Control rule acquisition for an arm wrestling robot". In: *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*. Vol. 5. IEEE. 4227–4231.

Kaplan, F., P.-Y. Oudeyer, E. Kubinyi, and A. Miklósi. (2002). "Robotic clicker training". *Robotics and Autonomous Systems*. 38(3): 197–206. DOI: https://doi.org/10.1016/S0921-8890(02)00168-9.

Ke, L., S. Choudhury, M. Barnes, W. Sun, G. Lee, and S. Srinivasa. (2020). "Imitation learning as f-divergence minimization". In: *International Workshop on the Algorithmic Foundations of Robotics*. Springer. 313–329.

Kelly, M., C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. (2019). "HG-DAgger: Interactive Imitation Learning with Human Experts". In: *2019 International Conference on Robotics and Automation (ICRA)*. 8077–8083. DOI: 10.1109/ICRA.2019.8793698.

Khetarpal, K., Z. Ahmed, G. Comanici, D. Abel, and D. Precup. (2020). "What can I do here? A Theory of Affordances in Reinforcement Learning". In: *International Conference on Machine Learning*. PMLR. 5243–5253.

Kimble, K., K. Van Wyk, J. Falco, E. Messina, Y. Sun, M. Shibata, W. Uemura, and Y. Yokokohji. (2020). "Benchmarking Protocols for Evaluating Small Parts Robotic Assembly Systems". *IEEE Robotics and Automation Letters*. 5(2): 883–889. DOI: 10.1109/LRA.2020.2965869.

Knox, W. B., C. Breazeal, and P. Stone. (2012). "Learning from feedback on actions past and intended". In: *In Proceedings of 7th ACM/IEEE International Conference on Human-Robot Interaction, Late-Breaking Reports Session (HRI 2012)*.

Knox, W. B. and P. Stone. (2008). "Tamer: Training an agent manually via evaluative reinforcement". In: *Development and Learning, 2008. ICDL 2008. 7th IEEE International Conference on*. IEEE. 292–297.

Knox, W. B. and P. Stone. (2010). "Combining manual feedback with subsequent MDP reward signals for reinforcement learning". In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 5–12.

Knox, W. B. and P. Stone. (2012). "Reinforcement learning from simultaneous human and MDP reward". In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 475–482.

Knox, W. B. and P. Stone. (2009). "Interactively Shaping Agents via Human Reinforcement: The TAMER Framework". In: *Proceedings of the Fifth International Conference on Knowledge Capture. K-CAP '09*. Redondo Beach, California, USA: Association for Computing Machinery. 9–16. DOI: 10.1145/1597735.1597738.

Knox, W. B. and P. Stone. (2013). "Learning Non-Myopically from Human-Generated Reward". In: *Proceedings of the 2013 International Conference on Intelligent User Interfaces. IUI '13*. Santa Monica, California, USA: Association for Computing Machinery. 191–202. DOI: 10.1145/2449396.2449422.

Knox, W. B. and P. Stone. (2015). "Framing reinforcement learning from human reward: Reward positivity, temporal discounting, episodicity, and performance". *Artificial Intelligence*. 225: 24–50. DOI: https://doi.org/10.1016/j.artint.2015.03.009.

Knox, W. B., P. Stone, and C. Breazeal. (2013). "Training a Robot via Human Feedback: A Case Study". In: *Proceedings of the 5th International Conference on Social Robotics - Volume 8239. ICSR 2013*. Bristol, UK: Springer-Verlag. 460–470. DOI: 10.1007/978-3-319-02675-6_46.

Kober, J. and J. Peters. (2008). "Policy Search for Motor Primitives in Robotics". 21. URL: https://proceedings.neurips.cc/paper/2008/file/7647966b7343c29048673252e490f736-Paper.pdf.

Koert, D., M. Kircher, V. Salikutluk, C. D'Eramo, and J. Peters. (2020). "Multi-Channel Interactive Reinforcement Learning for Sequential Tasks". *Frontiers in Robotics and AI*. 7. DOI: 10.3389/frobt.2020.00097.

Koert, D., J. Pajarinen, A. Schotschneider, S. Trick, C. Rothkopf, and J. Peters. (2019). "Learning intention aware online adaptation of movement primitives". *IEEE Robotics and Automation Letters*. 4(4): 3719–3726.

Koppol, P., H. Admoni, and R. Simmons. (2021). "Interaction considerations in learning from humans". In: *International Joint Conference on Artificial Intelligence (IJCAI)*.

Krening, S., B. Harrison, K. M. Feigh, C. L. Isbell, M. Riedl, and A. Thomaz. (2017). "Learning From Explanations Using Sentiment and Advice in RL". *IEEE Transactions on Cognitive and Developmental Systems*. 9(1): 44–55. DOI: 10.1109/TCDS.2016.2628365.

Kulak, T., H. Girgin, J.-M. Odobez, and S. Calinon. (2021). "Active learning of Bayesian probabilistic movement primitives". *IEEE Robotics and Automation Letters*. 6(2): 2163–2170.

Kulesza, T., S. Amershi, R. Caruana, D. Fisher, and D. Charles. (2014). "Structured labeling for facilitating concept evolution in machine learning". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 3075–3084.

Kuniavsky, M. (2003). *Observing the user experience: a practitioner's guide to user research*. Elsevier.

Laskey, M., C. Chuck, J. Lee, J. Mahler, S. Krishnan, K. Jamieson, A. Dragan, and K. Goldberg. (2017a). "Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 358–365. DOI: 10.1109/ICRA.2017.7989046.

Laskey, M., J. Lee, R. Fox, A. Dragan, and K. Goldberg. (2017b). "Dart: Noise injection for robust imitation learning". In: *Conference on robot learning*. PMLR. 143–156.

Laskey, M., S. Staszak, W. Y.-S. Hsieh, J. Mahler, F. T. Pokorny, A. D. Dragan, and K. Goldberg. (2016). "SHIV: Reducing supervisor burden in dagger using support vectors for efficient learning from demonstrations in high dimensional state spaces". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 462–469.

Le, H., N. Jiang, A. Agarwal, M. Dudik, Y. Yue, and H. Daumé III. (2018). "Hierarchical imitation and reinforcement learning". In: *International conference on machine learning*. PMLR. 2917–2926.

Lee, J. (2017). "A survey of robot learning from demonstrations for human-robot collaboration". *arXiv preprint arXiv:1710.08789*.

Lee, Y., E. S. Hu, Z. Yang, and J. J. Lim. (2020). "To Follow or not to Follow: Selective Imitation Learning from Observations". In: *Proceedings of the Conference on Robot Learning*. Vol. 100. *Proceedings of Machine Learning Research*. PMLR. 11–23. URL: https://proceedings.mlr.press/v100/lee20a.html.

León, A., E. F. Morales, L. Altamirano, and J. R. Ruiz. (2011). "Teaching a Robot to Perform Task through Imitation and On-Line Feedback". In: *Proceedings of the 16th Iberoamerican Congress Conference on Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. CIARP'11*. Pucón, Chile: Springer-Verlag. 549–556. DOI: 10.1007/978-3-642-25085-9_65.

Levine, S., A. Kumar, G. Tucker, and J. Fu. (2020). "Offline reinforcement learning: Tutorial, review, and perspectives on open problems". *arXiv preprint arXiv:2005.01643*.

Lewis, M. A., A. H. Fagg, A. Solidum, *et al.* (1992). "Genetic programming approach to the construction of a neural network for control of a walking robot." In: *ICRA*. Citeseer. 2618–2623.

Li, G., R. Gomez, K. Nakamura, and B. He. (2019a). "Human-centered reinforcement learning: A survey". *IEEE Transactions on Human-Machine Systems*. 49(4): 337–349.

Li, G., S. Whiteson, W. B. Knox, and H. Hung. (2016). "Using informative behavior to increase engagement while learning from human reward". *Autonomous agents and multi-agent systems*. 30(5): 826–848.

Li, G., M. Mueller, V. M. Casser, N. Smith, D. Michels, and B. Ghanem. (2019b). "OIL: Observational Imitation Learning". In: *Proceedings of Robotics: Science and Systems*. FreiburgimBreisgau, Germany. DOI: 10.15607/RSS.2019.XV.005.

Liese, F. and I. Vajda. (2006). "On divergences and informations in statistics and information theory". *IEEE Transactions on Information Theory*. 52(10): 4394–4412.

Lin, J., Z. Ma, R. Gomez, K. Nakamura, B. He, and G. Li. (2020). "A Review on Interactive Reinforcement Learning From Human Social Feedback". *IEEE Access*. 8: 120757–120765.

Lin, L.-J. (1992). "Self-Improving Reactive Agents Based on Reinforcement Learning, Planning and Teaching". *Machine Learning*. 8(3): 293–321. DOI: 10.1007/BF00992699.

Lin, Y., A. S. Wang, E. Undersander, and A. Rai. (2022). "Efficient and Interpretable Robot Manipulation With Graph Neural Networks". *IEEE Robotics and Automation Letters*. 7(2): 2740–2747. DOI: 10.1109/LRA.2022.3143518.

Loftin, R., J. MacGlashan, B. Peng, M. Taylor, M. Littman, J. Huang, and D. Roberts. (2014). "A Strategy-Aware Technique for Learning Behaviors from Discrete Human Feedback". In: vol. 28. No. 1. DOI: 10.1609/aaai.v28i1.8839.

Loftin, R., B. Peng, J. MacGlashan, M. L. Littman, M. E. Taylor, J. Huang, and D. L. Roberts. (2016). "Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning". *Autonomous agents and multi-agent systems*. 30(1): 30–59.

Londoño, L., A. Röfer, T. Welschehold, and A. Valada. (2022). "Doing Right by Not Doing Wrong in Human-Robot Collaboration". arXiv: 2202.02654 [cs.RO].

Losey, D. P., A. Bajcsy, M. K. O'Malley, and A. D. Dragan. (2022). "Physical interaction as communication: Learning robot objectives online from human corrections". *The International Journal of Robotics Research*. 41(1): 20–44. DOI: 10.1177/02783649211050958.

Lund, H., O. Miglino, L. Pagliarini, A. Billard, and A. Ijspeert. (1998). "Evolutionary robotics-a children's game". In: *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*. 154–158. DOI: 10.1109/ICEC.1998.699493.

Luo, J., O. Sushkov, R. Pevceviciute, W. Lian, C. Su, M. Vecerik, N. Ye, S. Schaal, and J. Scholz. (2021). "Robust Multi-Modal Policies for Industrial Assembly via Reinforcement Learning and Demonstrations: A Large-Scale Study". In: *Robotics: Science and Systems XVII, 2021*.

MacGlashan, J., M. K. Ho, R. Loftin, B. Peng, G. Wang, D. L. Roberts, M. E. Taylor, and M. L. Littman. (2017). "Interactive Learning from Policy-Dependent Human Feedback". In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. *Proceedings of Machine Learning Research*. PMLR. 2285–2294. URL: https://proceedings.mlr.press/v70/macglashan17a.html.

MacGlashan, J., M. Littman, R. Loftin, B. Peng, D. Roberts, and M. E. Taylor. (2014). "Training an agent to ground commands with reward and punishment". In: *Proceedings of the AAAI Machine Learning for Interactive Systems Workshop*. 6–12.

Maclin, R. and J. W. Shavlik. (1994). *Incorporating Advice into Agents That Learn from Reinforcements*. University of Wisconsin-Madison. Computer Sciences Department.

Maeda, G., M. Ewerton, T. Osa, B. Busch, and J. Peters. (2017). "Active Incremental Learning of Robot Movement Primitives". In: *Proceedings of the 1st Annual Conference on Robot Learning*. Vol. 78. *Proceedings of Machine Learning Research*. PMLR. 37–46. URL: https://proceedings.mlr.press/v78/maeda17a.html.

Mahmood, A. (2017). "Incremental off-policy reinforcement learning algorithms".

Mandlekar, A., J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei. (2019). "Scaling Robot Supervision to Hundreds of Hours with RoboTurk: Robotic Manipulation Dataset through Human Reasoning and Dexterity". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1048–1055. DOI: 10.1109/IROS40897.2019.8968114.

Mandlekar, A., D. Xu, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese. (2020). "Human-in-the-Loop Imitation Learning using Remote Teleoperation". DOI: 10.48550/ARXIV.2012.06733.

Mandlekar, A., Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, S. Savarese, and L. Fei-Fei. (2018). "ROBOTURK: A Crowdsourcing Platform for Robotic Skill Learning through Imitation". In: *Proceedings of The 2nd Conference on Robot Learning*. Vol. 87. *Proceedings of Machine Learning Research*. PMLR. 879–893. URL: https://proceedings.mlr.press/v87/mandlekar18a.html.

Marta, D., C. Pek, G. I. Melsión, J. Tumova, and I. Leite. (2021). "Human-Feedback Shield Synthesis for Perceived Safety in Deep Reinforcement Learning". *IEEE Robotics and Automation Letters*. 7(1): 406–413.

McCarthy, J. (1958). "Programs with Common Sense". In: RLE and MIT computation center Cambridge, MA, USA.

Menda, K., K. Driggs-Campbell, and M. J. Kochenderfer. (2017). "DropoutDAgger: A Bayesian Approach to Safe Imitation Learning". DOI: 10.48550/ARXIV.1709.06166.

Menda, K., K. Driggs-Campbell, and M. J. Kochenderfer. (2019). "EnsembleDAgger: A Bayesian Approach to Safe Imitation Learning". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 5041–5048. DOI: 10.1109/IROS40897.2019.8968287.

Mericli, C. (2011). "Multi-Resolution Model Plus Correction Paradigm for Task and Skill Refinement on Autonomous Robots". *PhD thesis*. Citeseer.

Meriçli, Ç. and M. Veloso. (2011). "Improving Biped Walk Stability Using Real-Time Corrective Human Feedback". In: *RoboCup 2010: Robot Soccer World Cup XIV*. Berlin, Heidelberg: Springer Berlin Heidelberg. 194–205.

Meriçli, Ç., M. Veloso, and H. L. Akin. (2010). "Complementary humanoid behavior shaping using corrective demonstration". In: *2010 10th IEEE-RAS International Conference on Humanoid Robots*. 334–339. DOI: 10.1109/ICHR.2010.5686326.

Meriçli, Ç., M. Veloso, and H. L. Akın. (2011). "Task Refinement for Autonomous Robots Using Complementary Corrective Human Feedback". *International Journal of Advanced Robotic Systems*. 8(2): 16. DOI: 10.5772/10575.

Mészáros, A., G. Franzese, and J. Kober. (2022). "Learning to Pick at Non-Zero-Velocity From Interactive Demonstrations". *IEEE Robotics and Automation Letters*. 7(3): 6052–6059. DOI: 10.1109/LRA.2022.3165531.

Mitsunaga, N., C. Smith, T. Kanda, H. Ishiguro, and N. Hagita. (2008). "Adapting robot behavior for human–robot interaction". *IEEE Transactions on Robotics*. 24(4): 911–916.

Mohseni, S., N. Zarei, and E. Ragan. (2019). "A Multidisciplinary survey and framework for design and evaluation of explainable AI systems. arXiv". *Human-Computer Interaction*.

Müller, M., V. Casser, J. Lahoud, N. Smith, and B. Ghanem. (2018). "Sim4cv: A photo-realistic simulator for computer vision applications". *International Journal of Computer Vision*. 126(9): 902–919.

Myers, V., E. Biyik, N. Anari, and D. Sadigh. (2022). "Learning Multimodal Rewards from Rankings". In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. *Proceedings of Machine Learning Research*. PMLR. 342–352. URL: https://proceedings.mlr.press/v164/myers22a.html.

Najar, A. and M. Chetouani. (2021). "Reinforcement Learning with Human Advice: A Survey". *Frontiers in Robotics and AI*. 8.

Najar, A., O. Sigaud, and M. Chetouani. (2016). "Training a robot with evaluative feedback and unlabeled guidance signals". In: *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on*. IEEE. 261–266.

Najar, A., O. Sigaud, and M. Chetouani. (2020). "Interactively shaping robot behaviour with unlabeled human instructions". *Autonomous Agents and Multi-Agent Systems*. 34(2): 1–35.

Nehaniv, C. L., K. Dautenhahn, *et al.* (2002). "The correspondence problem". *Imitation in animals and artifacts*. 41.

Ng, A. Y., D. Harada, and S. Russell. (1999). "Policy Invariance under Reward Transformations: Theory and Application to Reward Shaping". In: *Icml*. Vol. 99. 278–287.

Ng, A. Y. and S. J. Russell. (2000). "Algorithms for inverse reinforcement learning." In: *Icml*. 663–670.

Ngo, H., M. Luciw, J. Nagi, A. Forster, J. Schmidhuber, and N. A. Vien. (2014). "Efficient interactive multiclass learning from binary feedback". *ACM Transactions on Interactive Intelligent Systems (TiiS)*. 4(3): 12.

Nicolescu, M. N. and M. J. Mataric. (2003). "Natural Methods for Robot Task Learning: Instructive Demonstrations, Generalization and Practice". In: *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems. AAMAS '03*. Melbourne, Australia: Association for Computing Machinery. 241–248. DOI: 10.1145/860575.860614.

Nojima, Y., F. Kojima, and N. Kubota. (2003). "Trajectory generation for human-friendly behavior of partner robot using fuzzy evaluating interactive genetic algorithm". In: *Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for the New Millennium (Cat. No. 03EX694)*. Vol. 1. IEEE. 306–311.

Osa, T., J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. (2018). "An algorithmic perspective on imitation learning". *arXiv preprint arXiv:1811.06711*.

Palan, M., G. Shevchuk, N. Charles Landolfi, and D. Sadigh. (2019). "Learning Reward Functions by Integrating Human Demonstrations and Preferences". In: *Robotics: Science and Systems*.

Paraschos, A., C. Daniel, J. R. Peters, and G. Neumann. (2013). "Probabilistic movement primitives". In: *Advances in neural information processing systems*. 2616–2624.

Paull, L., J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang, *et al.* (2017). "Duckietown: an open, inexpensive and flexible platform for autonomy education and research". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 1497–1504.

Peng, B., J. MacGlashan, R. Loftin, M. L. Littman, D. L. Roberts, and M. E. Taylor. (2016). "A need for speed: Adapting agent action speed to improve task learning from non-expert humans". In: *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*.

Pérez-Dattari, R., B. Brito, O. de Groot, J. Kober, and J. Alonso-Mora. (2022). "Visually-guided motion planning for autonomous driving from interactive demonstrations". *Engineering Applications of Artificial Intelligence*. 116. DOI: https://doi.org/10.1016/j.engappai.2022.105277.

Pérez-Dattari, R., C. Celemin, G. Franzese, J. Ruiz-del-Solar, and J. Kober. (2020). "Interactive learning of temporal features for control: Shaping policies and state representations from human feedback". *IEEE Robotics & Automation Magazine*. 27(2): 46–54.

Pérez-Dattari, R., C. Celemin, J. Ruiz-del-Solar, and J. Kober. (2018). "Interactive learning with corrective feedback for policies based on deep neural networks". In: *International Symposium on Experimental Robotics*. Springer. 353–363.

Pérez-Dattari, R., C. Celemin, J. Ruiz-del-Solar, and J. Kober. (2019). "Continuous control for high-dimensional state spaces: An interactive learning approach". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 7611–7617.

Pilarski, P. M., M. R. Dawson, T. Degris, F. Fahimi, J. P. Carey, and R. S. Sutton. (2011). "Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning". In: *2011 IEEE International Conference on Rehabilitation Robotics*. 1–7. DOI: 10.1109/ICORR.2011.5975338.

Prakash, A., A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger. (2020). "Exploring data aggregation in policy learning for vision-based urban autonomous driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11763–11773. DOI: 10.1109/CVPR42600.2020.01178.

Precup, D. (2000). *Temporal abstraction in reinforcement learning*. University of Massachusetts Amherst.

Pryor, K. (1999). "Clicker training for dogs. Waltham, MA".

Rasmussen, C. E. and C. K. I. Williams. (2006). *Gaussian Processes for Machine Learning*. The MIT Press.

Ravichandar, H., A. S. Polydoros, S. Chernova, and A. Billard. (2020). "Recent advances in robot learning from demonstration". *Annual Review of Control, Robotics, and Autonomous Systems*. 3: 297–330.

Reddy, S., A. D. Dragan, and S. Levine. (2019). "SQIL: Imitation Learning via Regularized Behavioral Cloning". *CoRR*. abs/1905.11108. URL: http://arxiv.org/abs/1905.11108.

Rohmer, E., S. P. Singh, and M. Freese. (2013). "V-REP: A versatile and scalable robot simulation framework". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 1321–1326.

Ross, S. and J. A. Bagnell. (2014). "Reinforcement and imitation learning via interactive no-regret learning". *arXiv preprint arXiv:1406.5979*.

Ross, S. and D. Bagnell. (2010). "Efficient reductions for imitation learning". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 661–668.

Ross, S., G. Gordon, and D. Bagnell. (2011). "A reduction of imitation learning and structured prediction to no-regret online learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 627–635.

Rubinstein, R. Y. and D. P. Kroese. (2016). *Simulation and the Monte Carlo method*. Vol. 10. John Wiley & Sons.

Rummery, G. A. and M. Niranjan. (1994). *On-line Q-learning using connectionist systems*. Vol. 37. Citeseer.

Russell, S. J. and P. Norvig. (2016). *Artificial Intelligence : A Modern Approach*. Malaysia and Pearson Education Limited.

Sadigh, D., A. Dragan, S. Sastry, and S. Seshia. (2017). "Active Preference-Based Learning of Reward Functions". In: *Proceedings of Robotics: Science and Systems*. Cambridge, Massachusetts. DOI: 10.15607/RSS.2017.XIII.053.

Samuel, A. L. (1959). "Some Studies in Machine Learning Using the Game of Checkers". *IBM Journal of Research and Development*. 3(3): 210–229. DOI: 10.1147/rd.33.0210.

Samuel, A. L. (1967). "Some Studies in Machine Learning Using the Game of Checkers. II–Recent Progress". *IBM Journal of Research and Development*. 11(6): 601–617. DOI: 10.1147/rd.116.0601.

Saran, A., R. Zhang, E. S. Short, and S. Niekum. (2021). "Efficiently Guiding Imitation Learning Agents with Human Gaze". In: *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*. 1109–1117.

Saveriano, M., F. J. Abu-Dakka, A. Kramberger, and L. Peternel. (2021). "Dynamic Movement Primitives in Robotics: A Tutorial Survey". DOI: 10.48550/ARXIV.2102.03861.

Savva, M., A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra. (2019). "Habitat: A Platform for Embodied AI Research". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Scholten, J., D. Wout, C. Celemin, and J. Kober. (2019). "Deep Reinforcement Learning with Feedback-based Exploration". In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE. 803–808.

Schroecker, Y., H. Ben Amor, and A. Thomaz. (2016). "Directing Policy Search with Interactively Taught Via-Points". In: *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems. AAMAS '16*. Singapore, Singapore: International Foundation for Autonomous Agents and Multiagent Systems. 1052–1059.

Schulman, J., Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel. (2014). "Motion planning with sequential convex optimization and convex collision checking". *The International Journal of Robotics Research*. 33(9): 1251–1270.

Settles, B. (2009). "Active learning literature survey".

Shah, S., D. Dey, C. Lovett, and A. Kapoor. (2018). "Airsim: High-fidelity visual and physical simulation for autonomous vehicles". In: *Field and service robotics*. Springer. 621–635.

Shridhar, M., D. Mittal, and D. Hsu. (2020). "INGRESS: Interactive visual grounding of referring expressions". *The International Journal of Robotics Research*. 39(2-3): 217–232. DOI: 10.1177/0278364919897133.

Silver, D., A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. (2016). "Mastering the Game of Go with Deep Neural Networks and Tree Search". *Nature*. 529(7587): 484–489. DOI: 10.1038/nature16961.

Sinha, S., A. Mandlekar, and A. Garg. (2022). "S4RL: Surprisingly Simple Self-Supervision for Offline Reinforcement Learning in Robotics". In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. *Proceedings of Machine Learning Research*. PMLR. 907–917. URL: https://proceedings.mlr.press/v164/sinha22a.html.

Smith, J. R. (1991). "Designing biomorphs with an interactive genetic algorithm." In: *ICGA*. Citeseer. 535–538.

Spencer, J., S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa. (2020). "Learning from interventions". In: *Robotics: Science and Systems (RSS)*.

Sperrle, F., M. El-Assady, G. Guo, R. Borgo, D. H. Chau, A. Endert, and D. Keim. (2021). "A Survey of Human-Centered Evaluations in Human-Centered Machine Learning". In: *Computer Graphics Forum*. Vol. 40. No. 3. Wiley Online Library. 543–568.

Sridharan, M. (2011). "Augmented reinforcement learning for interaction with non-expert humans in agent domains". In: *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*. Vol. 1. IEEE. 424–429.

Stapelberg, B. and K. M. Malan. (2020). "A survey of benchmarks for reinforcement learning algorithms". *South African Computer Journal*. 32(2).

Stulp, F. and O. Sigaud. (2015). "Many regression algorithms, one unified model: A review". *Neural Networks*. 69: 60–79.

Suay, H. B. and S. Chernova. (2011). "Effect of human guidance and state space size on interactive reinforcement learning". In: *RO-MAN, 2011 IEEE*. IEEE. 1–6.

Subramanian, K., C. L. Isbell Jr, and A. L. Thomaz. (2016). "Exploration from demonstration for interactive reinforcement learning". In: *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. 447–456.

Sugiyama, M. (2015). *Introduction to statistical machine learning*. Morgan Kaufmann.

Sun, W., A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell. (2017). "Deeply AggreVaTeD: Differentiable Imitation Learning for Sequential Prediction". In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. *Proceedings of Machine Learning Research*. PMLR. 3309–3318. URL: https://proceedings.mlr.press/v70/sun17d.html.

Sutton, R. S. and A. G. Barto. (2018). *Reinforcement learning: An introduction*. MIT press.

Szot, A., A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. M. Turner, N. D. Maestre, M. Mukadam, D. S. Chaplot, O. Maksymets, *et al.* (2021). "Habitat 2.0: Training Home Assistants to Rearrange their Habitat". In: *Thirty-Fifth Conference on Neural Information Processing Systems*.

Takagi, H. (1998). "Interactive evolutionary computation". In: *Proceedings of the International Conference on Soft Computing and Information/Intelligent Systems*. 41–50.

Takagi, H. (2001). "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation". *Proceedings of the IEEE*. 89(9): 1275–1296.

Tenorio-Gonzalez, A. C., E. F. Morales, and L. Villaseñor-Pineda. (2010). "Dynamic Reward Shaping: Training a Robot by Voice". In: *Advances in Artificial Intelligence – IBERAMIA 2010*. Berlin, Heidelberg: Springer Berlin Heidelberg. 483–492.

Thomaz, A. and C. Breazeal. (2006). "Adding guidance to interactive reinforcement learning". In: *Proceedings of the Twentieth Conference on Artificial Intelligence (AAAI)*.

Thomaz, A. L. and C. Breazeal. (2007a). "Asymmetric Interpretations of Positive and Negative Human Feedback for a Social Learning Agent". In: *RO-MAN 2007 - The 16th IEEE International Symposium on Robot and Human Interactive Communication*. 720–725. DOI: 10.1109/ROMAN.2007.4415180.

Thomaz, A. L. and C. Breazeal. (2007b). "Robot learning via socially guided exploration": 82–87. DOI: 10.1109/DEVLRN.2007.4354078.

Thomaz, A. L. and M. Cakmak. (2009). "Learning about objects with human teachers". In: *2009 4th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 15–22. DOI: 10.1145/1514095.1514101.

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

Thomaz, A. L., G. Hoffman, and C. Breazeal. (2005). "Real-Time Interactive Reinforcement Learning for Robots". In: *AAAI 2005 Workshop on Human Comprehensible Machine Learning*. 9–13.

Todorov, E., T. Erez, and Y. Tassa. (2012). "MuJoCo: A physics engine for model-based control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 5026–5033. DOI: 10.1109/IROS.2012.6386109.

Torabi, F., G. Warnell, and P. Stone. (2018). "Behavioral cloning from observation". *arXiv preprint arXiv:1805.01954*.

Toris, R., H. B. Suay, and S. Chernova. (2012). "A practical comparison of three robot learning from demonstration algorithms". In: *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE. 261–262.

Utgoff, P. (1991). "Two Kinds of Training Information for Evaluation Function Learning". *Computer Science Department Faculty Publication Series*: 193.

Van Der Laan, J. D., A. Heino, and D. De Waard. (1997). "A simple procedure for the assessment of acceptance of advanced transport telematics". *Transportation Research Part C: Emerging Technologies*. 5(1): 1–10.

Vecerik, M., T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller. (2017). "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards". *arXiv preprint arXiv:1707.08817*.

Vien, N. A. and W. Ertel. (2012). "Reinforcement learning combined with human feedback in continuous state and action spaces". In: *2012 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL)*. 1–6. DOI: 10.1109/DevLrn.2012.6400849.

Vien, N. A., W. Ertel, and T. C. Chung. (2013). "Learning via human feedback in continuous state and action spaces". *Applied intelligence*. 39(2): 267–278.

Vollmer, A.-L. and N. J. Hemion. (2018). "A user study on robot skill learning without a cost function: Optimization of dynamic movement primitives via naive user feedback". *Frontiers in Robotics and AI*. 5: 77.

Ware, M., E. Frank, G. Holmes, M. Hall, and I. H. Witten. (2001). "Interactive machine learning: letting users build classifiers". *International Journal of Human-Computer Studies*. 55(3): 281–292.

Warnell, G., N. Waytowich, V. Lawhern, and P. Stone. (2018). "Deep tamer: Interactive agent shaping in high-dimensional state spaces". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. No. 1.

Watkins, C. J. and P. Dayan. (1992). "Q-learning". *Machine learning*. 8(3): 279–292.

Watkins, C. J. C. H. (1989). "Learning from delayed rewards".

Wenskovitch, J. and C. North. (2020). "Interactive Artificial Intelligence: Designing for the" Two Black Boxes" Problem". *Computer*. 53(8): 29–39.

Whitehead, S. D. (1991). "A Complexity Analysis of Cooperative Mechanisms in Reinforcement Learning." In: *AAAI*. 607–613.

Wilde, N., E. Biyik, D. Sadigh, and S. L. Smith. (2022). "Learning Reward Functions from Scale Feedback". In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. *Proceedings of Machine Learning Research*. PMLR. 353–362. URL: https://proceedings.mlr.press/v164/wilde22a.html.

Wilde, N., A. Blidaru, S. L. Smith, and D. Kulić. (2020). "Improving user specifications for robot behavior through active preference learning: Framework and evaluation". *The International Journal of Robotics Research*. 39(6): 651–667. DOI: 10.1177/0278364920910802.

Williams, E. C., N. Gopalan, M. Rhee, and S. Tellex. (2018). "Learning to Parse Natural Language to Grounded Reward Functions with Weak Supervision". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 4430–4436. DOI: 10.1109/ICRA.2018.8460937.

Wilson, A., A. Fern, and P. Tadepalli. (2012). "A Bayesian Approach for Policy Learning from Trajectory Preference Queries". In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1. NIPS'12*. Lake Tahoe, Nevada: Curran Associates Inc. 1133–1141.

Wout, D., J. Scholten, C. Celemin, and J. Kober. (2019). "Learning Gaussian Policies from Corrective Human Feedback". DOI: 10.48550/ARXIV.1903.05216.

Wrede, S., C. Emmerich, R. Grünberg, A. Nordmann, A. Swadzba, and J. Steil. (2013). "A User Study on Kinesthetic Teaching of Redundant Robots in Task and Configuration Space". *J. Hum.-Robot Interact.* 2(1): 56–81. DOI: 10.5898/JHRI.2.1.Wrede.

Wu, X., L. Xiao, Y. Sun, J. Zhang, T. Ma, and L. He. (2021). "A Survey of Human-in-the-loop for Machine Learning". *arXiv preprint arXiv:2108.00941*.

Wulfmeier, M., D. Z. Wang, and I. Posner. (2016). "Watch this: Scalable cost-function learning for path planning in urban environments". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2089–2095.

Xiao, B., Q. Lu, B. Ramasubramanian, A. Clark, L. Bushnell, and R. Poovendran. (2020). "FRESH: Interactive Reward Shaping in High-Dimensional State Spaces Using Human Feedback". In: *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems. AAMAS '20*. Auckland, New Zealand: International Foundation for Autonomous Agents and Multiagent Systems. 1512–1520.

Xin, D., L. Ma, J. Liu, S. Macke, S. Song, and A. Parameswaran. (2018). "Accelerating human-in-the-loop machine learning: Challenges and opportunities". In: *Proceedings of the second workshop on data management for end-to-end machine learning*. 1–4.

Yang, S., W. Zhang, W. Lu, H. Wang, and Y. Li. (2019). "Learning Actions from Human Demonstration Video for Robotic Manipulation". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1805–1811. DOI: 10.1109/IROS40897.2019.8968278.

Yanik, P. M., J. Manganelli, J. Merino, A. L. Threatt, J. O. Brooks, K. E. Green, and I. D. Walker. (2014). "A Gesture Learning Interface for Simulated Robot Path Shaping With a Human Teacher". *IEEE Transactions on Human-Machine Systems*. 44(1): 41–54. DOI: 10.1109/TSMC.2013.2291714.

Yu, T., D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. (2019). "Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning". In: *Conference on Robot Learning (CoRL)*. URL: https://arxiv.org/abs/1910.10897.

Zanzotto, F. M. (2019). "Human-in-the-loop artificial intelligence". *Journal of Artificial Intelligence Research*. 64: 243–252.

Zhang, J. and K. Cho. (2016). "Query-Efficient Imitation Learning for End-to-End Autonomous Driving". DOI: 10.48550/ARXIV.1605.06450.

Zhang, Q., J. Lin, Q. Sha, B. He, and G. Li. (2020). "Deep Interactive Reinforcement Learning for Path Following of Autonomous Underwater Vehicle". *IEEE Access*. 8: 24258–24268. DOI: 10.1109/ACCESS.2020.2970433.

Zhang, Q., L. Zha, J. Lin, D. Tu, M. Li, F. Liang, R. Wu, and X. Lu. (2019a). "A Survey on Deep Learning Benchmarks: Do We Still Need New Ones?" In: *Benchmarking, Measuring, and Optimizing*. Cham: Springer International Publishing. 36–49.

Zhang, R., F. Torabi, L. Guan, D. H. Ballard, and P. Stone. (2019b). "Leveraging human guidance for deep reinforcement learning tasks". *arXiv preprint arXiv:1909.09906*.

Zhifei, S. and E. M. Joo. (2012). "A review of inverse reinforcement learning theory and recent advances". In: *Evolutionary Computation (CEC), 2012 IEEE Congress on*. IEEE. 1–8.

Zhu, Y., J. Wong, A. Mandlekar, and R. Martín-Martín. (2020). "robosuite: A Modular Simulation Framework and Benchmark for Robot Learning". arXiv: 2009.12293 [cs.RO].