A Survey of Robot Learning from Demonstration

Brenna D. Argall, Sonia Chernova, Manuela Veloso and Brett Browning

Abstract

We present a comprehensive survey of robot *Learning from Demonstration (LfD)*, a technique that develops policies from example state to action mappings. We introduce the LfD design choices in terms of demonstrator, problem space, policy derivation and performance, and contribute the foundations for a structure in which to categorize LfD research. Specifically, we analyze and categorize the multiple ways in which examples are gathered, ranging from teleoperation to imitation, as well as the various techniques for policy derivation, including matching functions, dynamics models and plans. To conclude we discuss LfD limitations and related promising areas for future research.

Keywords: Learning from Demonstration, Robotics, Machine Learning, Autonomous Systems

1 Introduction

The problem of learning a mapping between world state and actions lies at the heart of many robotics applications. This mapping, also called a *policy*, enables a robot to select an action based upon its current world state. The development of policies by hand is often a very challenging problem, and as a result many machine learning techniques have been applied to policy development. In this survey, we examine a particular approach to policy learning, *Learning from Demonstration (LfD)*.

Within LfD, a policy is learned from *examples*, or demonstrations, provided by a teacher. We define examples as sequences of state-action pairs that are recorded during the teacher's demonstration of the desired robot behavior. LfD algorithms utilize this dataset of examples to derive a policy that reproduces the demonstrated behavior. This approach to obtaining a policy is in contrast to other techniques in which a policy is learned from *experience*, for example building a policy based on data acquired through exploration, as in Reinforcement Learning [Sutton and Barto, 1998]. We note that a policy derived under LfD is necessarily defined only in those states encountered, and for those actions taken, during the example executions.

In this article, we present a survey of recent work within the LfD community, focusing specifically on robotic applications. We segment the LfD learning problem into two fundamental phases: *gathering* the examples, and *deriving* a policy from them. Based on our identification of the defining features of these techniques, we contribute a comprehensive survey and categorization of existing LfD approaches. Though LfD has been applied to a variety of robotics problems, to our knowledge there exists *no* established structure for concretely placing work within the larger community. In general, approaches are appropriately contrasted to similar or seminal research, but their relation to the remainder of the field lies largely unaddressed. Establishing these relations is further complicated by dealing with real world robotic platforms, for which the physical details between implementations may vary greatly and yet employ fundamentally identical learning techniques, or vice versa. A categorical structure therefore aids in comparative assessments between applications, as well as in identifying open areas for future research. In contributing our categorization of current approaches, we aim to lay the foundations for such a structure.

For the remainder of this section we motivate the application of LfD to robotics, and present a formal definition of the LfD problem. Section 2 presents the key design decisions for an LfD system. Methods for gathering demonstration examples are the focus of Section 3, where the various approaches to teacher demonstration and data recording are discussed. Section 4 examines the core techniques for policy derivation within LfD, followed in Section 5 by methods for improving robot performance beyond the capabilities of the teacher examples. To conclude, we identify and discuss open areas of research for future work in Section 6 and summarize with Section 7.

1.1 Support for Demonstration Learning

The presence of robots within society is becoming ever more prevalent. Whether an exploration rover in space or recreational robot for the home, successful autonomous robot operation requires robust control algorithms. Non-robotics-experts are increasingly presented with opportunities to interact with robots, and it is reasonable to expect that they have ideas about what a robot should do, and therefore what sort of behaviors these control algorithms should produce. A natural, and practical, extension of having this knowledge is to actually develop the desired control algorithm. Currently, however, policy development is a complex process restricted to experts within the field.

Traditional approaches to robot control model domain dynamics and derive mathematically-based policies. Though theoretically well-founded, these approaches depend heavily upon the accuracy of the world model. Not only does this model require considerable expertise to develop, but approximations such as linearization are often introduced for computational tractability, thereby degrading performance. Other approaches, such as Reinforcement Learning, guide policy learning by providing reward feedback about the desirability of visiting particular states. To define a function to provide these rewards, however, is known to be a difficult problem and also requires considerable expertise to address. Furthermore, building the policy requires gathering information by visiting states to receive rewards, which is non-trivial for a robot learner executing actual actions in the real world.

Considering these challenges, LfD has many attractive points for both learner and teacher. LfD formulations typically do not require expert knowledge of the domain dynamics, which removes performance brittleness resulting from model simplifications. This also opens policy development to non-robotics-experts, satisfying a need which increases as robots become more commonplace. Furthermore, demonstration has the attractive feature of being an intuitive medium for communication from humans, who already use demonstration to teach other humans. Demonstration also has the practical feature of focusing the dataset to areas of the state-space actually encountered during task execution.

1.2 Problem Statement

We formally construct the LfD problem as follows. The world consists of states S and actions A, with the mapping between states by way of actions being defined by a probabilistic transition function $T(s'|s,a) : S \times A \times S \rightarrow [0,1]$. We assume that state is not fully observable. The learner instead has access to observed state Z, through the mapping $M : S \rightarrow Z$. A policy $\pi : Z \rightarrow A$ selects actions based on observations of the world state. A single cycle of policy execution at time t is shown in Figure 1 (bottom).

The set A ranges from containing low-level motions to high-level behaviors. For some simulated world applications, state may be fully transparent, in which case $M = I$, the identity mapping. For all other applications state is not fully transparent and must be observed, for example through sensors in the real world. For succinctness, throughout the text we will use "state" interchangeably with "observed state." It should be assumed, however, that state is always the observed state, unless explicitly noted otherwise. This assumption will be reinforced by use of the Z notation throughout the text.

LfD can be seen as a subset of *Supervised Learning*. In Supervised Learning the agent is presented with training data and learns an approximation to the function which produced the data. Within LfD, this training dataset is composed of example executions of the task by a demonstration teacher (Fig. 1, top). Throughout the teacher execution, states and selected actions are recorded. We represent a demonstration $d_j \in D$ formally as k_j pairs of observations and actions: $d_j = \{(z^i_j, a^i_j)\},\ z^i_j \in Z,\ a^i_j \in A,\ i = 0 \cdots k_j$. What sets LfD apart from other learning approaches is the existence of these demonstrations. The set D of the demonstrations is made available to the learner. The policy derived from this dataset enables the learner to select an action based on the current state.

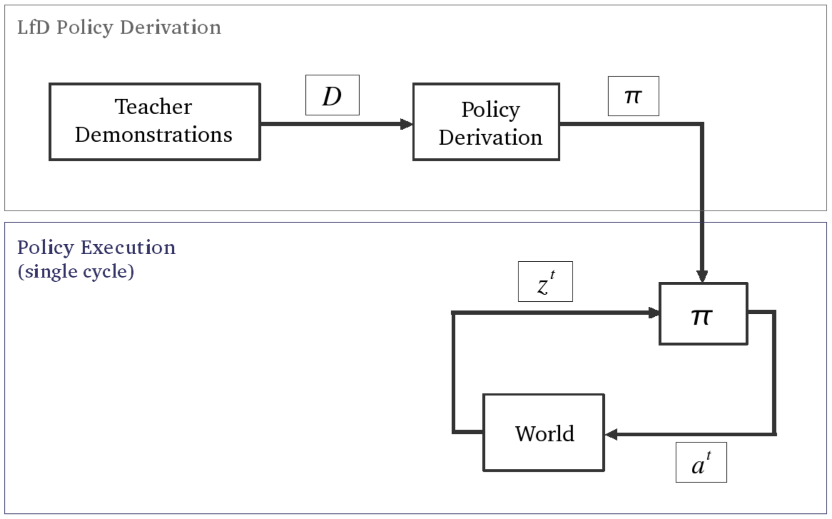

Figure 1: Control policy derivation and execution.

1.3 Terminology and Context

Before continuing, we pause to place the intents of this survey within the context of previous LfD literature. The aim of this survey is to review the broad topic of Learning from Demonstration, to provide a categorization that highlights differences between approaches, and to identify research areas within LfD that have not yet been explored.

We begin with a comment on terminology. Demonstration-based learning techniques are described by a variety of terms within the published literature, including Learning by Demonstration (LbD), Learning from Demonstration (LfD), Programming by Demonstration (PbD), Learning by Experienced Demonstrations, Assembly Plan from Observation, Learning by Showing, Learning by Watching, Learning from Observation, behavioral cloning, imitation and mimicry. While the definitions for some of these terms, such as imitation, have been loosely borrowed from other sciences, the overall use of these terms is often inconsistent or contradictory across articles.

Within this article, we refer to the general category of algorithms in which a policy is derived based on demonstrated data as *Learning from Demonstration (LfD)*. Within this category, we further distinguish between approaches by their various characteristics, as outlined in Section 2, such as the source of the demonstration and the learning techniques applied. Terms used to characterize algorithmic differences will be introduced in subsequent sections. Due to the already contradictory use of terms in existing literature, our definitions will not always agree with those of other publications. Our intent, however, is not for others in the field to adopt the terminology presented here, but rather to provide a consistent set of definitions that highlight distinctions between techniques.

Regarding a categorization for approaches, we note that many legitimate criteria could be used to subdivide LfD research. For example, one proposed categorization considers the broad spectrum of *who, what, when and how to imitate*, or subsets thereof [Billard et al., 2008, Schaal et al., 2003]. Our review aims to focus on the specifics of implementation. We therefore categorize approaches according to the computational formulations and techniques required to implement an LfD system.

To conclude, readers may also find useful other related surveys of the LfD research area. In particular, the book *Imitation in Animals and Artifacts* by Dautenhahn and Nehaniv [2002] provides an interdisciplinary overview of research in imitation learning, presenting leading work from neuroscience, psychology and linguistics as well as computer science. A narrower focus is presented in the chapter "Robot Programming by Demonstration" [Billard et al., 2008] within the book *Handbook of Robotics*. This work particularly highlights techniques which may augment or combine with traditional LfD, such as giving the teacher an active role during learning. By contrast, our focus, in addition to presenting implementation specifics, is to provide a categorical structure for LfD approaches. We do refer the reader to this chapter for a more comprehensive historical overview of LfD, as the scope of our survey is restricted to recently published literature. Additional reviews that cover specific sub-areas of LfD research in detail will be highlighted throughout the text.

2 Design Choices

There are certain aspects of LfD which are common among all applications to date. One is the fact that a teacher demonstrates execution of a desired behavior. Another is that the learner is provided with a set of these demonstrations, and from them derives a policy able to reproduce the demonstrated behavior.

However, the developer still faces many design choices when developing a new LfD system. Some of these decisions, such as the choice of a discrete or continuous action representation, may be determined by the domain. Other design choices may be up to the preference of the programmer. As we discuss in the later sections, these design decisions strongly influence how the learning problem is structured and solved. In this section we highlight those decisions that are the most significant to make.

To illustrate these choices, we will pair the presentation with a running *pick and place* example in which a robot must move a box from a table to a chair. To do so, the object must be 1) picked up, 2) relocated and 3) put down. We will present alternate representations and/or learning methods for this task, to illustrate how these particular choices influence task formalization and learning.

2.1 Demonstration Approach

Within the context of gathering teacher demonstrations, two key decisions must be made: the choice of *demonstrator*, and the choice of *demonstration technique*. We discuss various choices for each of these decisions below. Note that these decisions are at times effected by factors such as the complexity of the robot and task. For example, teleoperation is rarely used with high degree of freedom humanoids, since their complex motions are typically difficult to control via joystick.

2.1.1 The Choice of Demonstrator

Most LfD work to date has made use of human demonstrators, although some techniques have also examined the use of robotic teachers, hand-written control policies and simulated planners. The choice of demonstrator further breaks down into the subcategories of (i) who *controls* the demonstration and (ii) who *executes* the demonstration.

For example, consider a robot learning to move a box, as described above. One demonstration approach could have a robotic teacher pick up and relocate the box using its own body. In this case a *robot* teacher controls the demonstration, and its *teacher* body executes the demonstration. An alternate approach could have a human teacher teleoperate the robot learner through the task of picking up and relocating the box. In this case a *human* teacher controls the demonstration, and the *learner* body executes the demonstration. The choice of demonstrator has a significant impact on the type of learning algorithms that can be applied. As we discuss in Section 3, the similarity between the state and action spaces of the teacher and learner determines the kinds of algorithms that may be required to process the data.

2.1.2 Demonstration Technique

The choice of demonstration technique refers to the *strategy* for providing data to the learner. One option is to perform batch learning, in which case the policy is learned only once all data has been gathered. Alternatively, interactive approaches allow the policy to be updated incrementally as training data becomes available, possibly provided in response to current policy performance. Examples of both approaches will be highlighted throughout the article.

2.2 Problem Space Continuity

The question of continuity plays a prominent role within the context of state and action representation, and many valid representations frequently exist for the same domain. Within our robot box moving example, one option could be to discretize state, such that the environment is represented by boolean features such as *box on table* and *box held by robot*. Alternatively, a continuous state representation could be used in which the state is represented by the 3-D positions of the robot's end effector and the box. Similar discrete or continuous representations could be chosen for the robot's actions.

In designing a domain, the continuity of the problem space may be determined by many factors, such as the desired learned behavior, the set of available actions and whether the world is simulated or real. As discussed in Section 4, the question of continuity heavily influences how suitable the various policy derivation techniques are for addressing a given problem.

Additionally, we note that LfD can be applied at a variety of action control levels, depending on the problem formulation. We roughly group actions into three control levels: low level actions for motion control, basic high level actions (often called action primitives) and complex behavioral actions for high level control. Note that this is a somewhat different consideration to action-space continuity. For example, a low level motion could be formulated as discrete or continuous, and so this action level can map to either space. As a general technique, LfD can be applied at any of these action levels. Most important in the context of policy derivation, however, is whether actions are continuous or discrete, and not their control level. For the remainder of the paper, we therefore distinguish between representations based on continuity only.

2.3 Policy Derivation and Performance

In selecting an algorithm for generating a policy, two key decisions must be made: the general technique used to *derive the policy*, and whether performance can *improve beyond the teacher's demonstrations*. These decisions are influenced by action-continuity, as described in the previous section, which is in turn determined both by task and robot capabilities.

2.3.1 Policy Derivation Technique

As summarized in Figure 2, research within LfD has seen the development of three core approaches to policy derivation from demonstration data, which we define as *mapping function*, *system model*, and *plans*:

- *Mapping Function* (Section 4.1) : Demonstration data is used to directly approximate the underlying function mapping from the robot's state observations to actions ($f() : Z \rightarrow A$).
- *System Model* (Section 4.2) : Demonstration data is used to determine a model of the world dynamics ($T(s'|s, a)$), and possibly a reward function ($R(s)$). A policy is then derived using this information.
- *Plans* (Section 4.3) : Demonstration data, and often additional user intention information, is used to learn rules that associate a set of pre- and post-conditions with each action ($L(\{preC, postC\}|a)$), and possibly a sparsified state dynamics model ($T(s'|s, a)$). A sequence of actions is then planned using this information.

Returning again to our example, suppose a mapping function approach is used to derive the policy. A function $f() : Z \rightarrow A$ is learned that maps the observed state of the world, for example the 3-D location of the robot's end effector, to an action which guides the learner towards the goal state, for example the desired end effector motor speed. Consider instead using a system model approach. Here a state transition model $T(s'|s, a)$ is learned, for example that taking the *pick up* action when in state *box on table* results in state *box held by robot*. Using this model, a policy is derived which indicates the best action to take when in a given state, such that the robot is guided towards the goal state. Finally, consider using a planning approach. The pre- and post-conditions of executing an action $L(\{preC, postC\}|a)$ are learned from the demonstrations. For example, the *pick up* action requires the *box on table* pre-condition, and results in the *box held by robot* post-condition. A planner uses this learned information to produce a sequence of actions that end with the robot in the goal state. Each of the three approaches will be discussed in detail within Section 4.

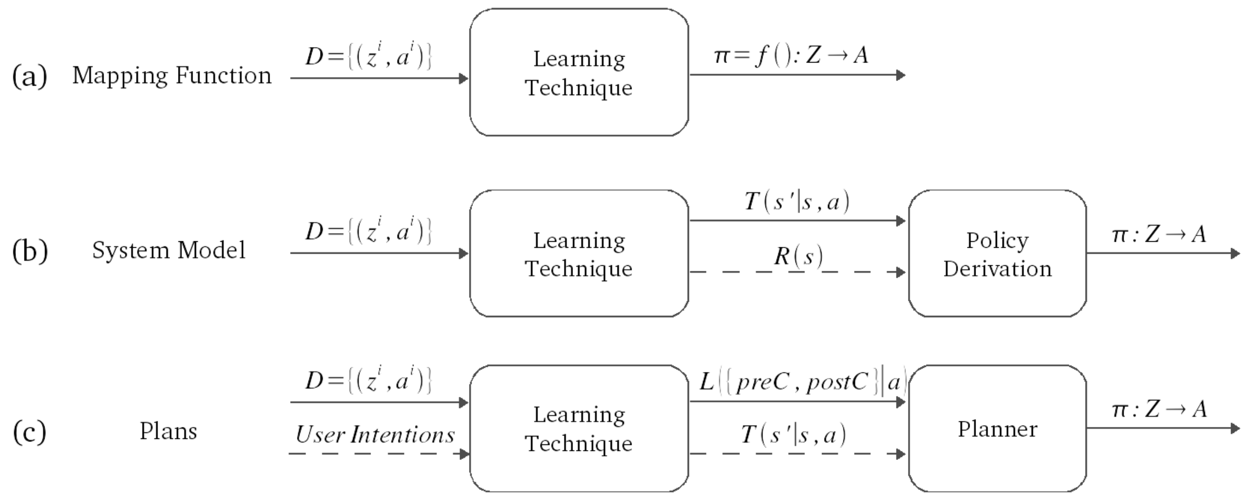

Figure 2: Policy derivation using the generalization approach of determining (a) an approximation to the state→action mapping function, (b) a dynamics model of the system and (c) a plan of sequenced actions.

2.3.2 Dataset Limitations

Training examples obtained from demonstration are inherently limited by the performance of the teacher. In many domains, it is possible that this teacher performance is suboptimal when compared with the abilities of the learner. For example, a human teacher may not be physically able to execute actions as quickly or accurately as a robot. Since the learner derives its policy from these examples, the performance of this policy is therefore also limited by the teacher's abilities. Many LfD learning systems, however, have been augmented to enable learner performance to improve beyond what was provided in the demonstration dataset. Examples include the incorporation of teacher advice or Reinforcement Learning techniques. These approaches will be discussed in depth within Section 5.

3 Gathering Examples: How the Dataset Is Built

In this section, we discuss various techniques for executing and recording demonstrations. The LfD dataset is composed of state-action pairs recorded during teacher executions of the desired behavior. Exactly *how* they are recorded, and *what* the teacher uses as a platform for the execution, varies greatly across approaches. Examples range from sensors on the robot learner recording its own actions as it is passively teleoperated by the teacher, to a camera recording a human teacher as she executes the behavior with her own body.

For LfD to be successful, the states and actions in the learning dataset must be usable by the student. In the most straightforward setup, the states and actions of the teacher executions map directly to the learner. In reality, however, this will often not be possible, as the learner and teacher will likely differ in sensing or mechanics. For example, a robot learner's camera will not detect state changes in the same manner as a human teacher's eyes, nor will its gripper apply force in the same manner as a human hand. The challenges which arise from these differences are referred to broadly as *Correspondence Issues* [Nehaniv and Dautenhahn, 2002].

3.1 Correspondence

The issue of correspondence deals with the identification of a mapping between the teacher and the learner which allows the transfer of information from one to the other. In this survey, we define correspondence with respect to two mappings, shown in Figure 3: the *record mapping*, and the *embodiment mapping*.

- The *Record Mapping* (Teacher Execution $\rightarrow$ Recorded Execution), refers to whether the exact states/actions experienced by the teacher during the demonstration execution are recorded within the dataset.
- The *Embodiment Mapping* (Recorded Execution $\rightarrow$ Learner), refers to whether the states/actions recorded within the dataset are exactly those that the learner would observe/execute.

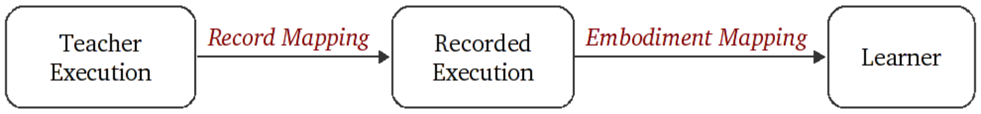

Figure 3: Mapping a teacher execution to the learner.

When the *record mapping* is the identity $I(z, a)$, the states/actions experienced by the teacher during execution are directly recorded in the dataset. Otherwise this teacher information is encoded according to some record mapping function $g_R(z, a) \neq I(z, a)$, and this encoded information is recorded within the dataset. Similarly, when the *embodiment mapping* is the identity $I(z, a)$, the states/actions in the dataset map directly to the learner. Otherwise the embodiment mapping consists of some function $g_E(z, a) \neq I(z, a)$. For any given learning system, it is possible to have neither, either or both of the record and embodiment mappings be the identity. Note that the mappings do not change the content of the demonstration data, but only the reference frame within which it is represented. The intersection of these configurations is shown in Figure 4, which will be discussed further within subsequent sections.

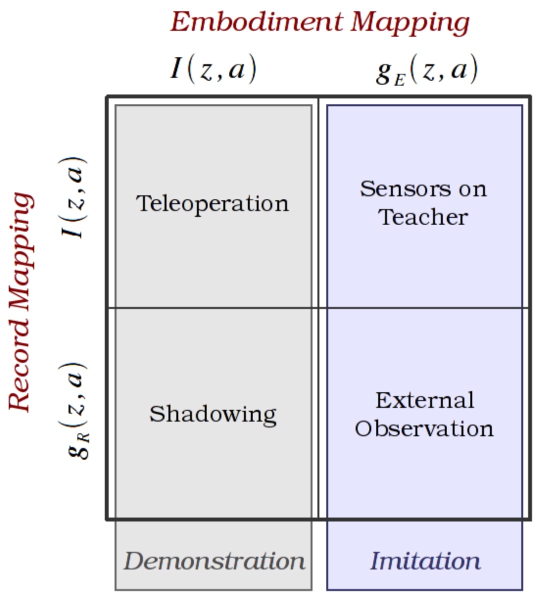

Figure 4: Intersection of the record and embodiment mappings. The left and right columns represent an identity (Demonstration) and non-identity (Imitation) embodiment mapping, respectively. Each column is then subdivided by an identity (top) or non-identity (bottom) record mapping. Typical approaches to providing data are listed within the quadrants.

The embodiment mapping is particularly important when considering real robots, compared with simulated agents. Since actual robots execute real actions within a physical environment, providing them with demonstration involves a physical execution by the teacher. Learning within this setting is heavily dependent upon an accurate mapping between the recorded dataset and the learner's abilities.

Recalling again our box relocation example, consider a human teacher using her own body to demonstrate moving the box, and that the demonstration is recorded by a camera. Let the teacher actions, A_T, be represented as human joint angles, and the learner actions, A_L, be represented as robot joint angles. In this context, the teacher's demonstration of the task is observed by the robot through the camera images. The teacher's exact actions are unknown to the robot; instead, this information must be extracted from the image data. This is an example of a $g_R(z, a) \neq I(z, a)$ *record* mapping $A_T \rightarrow D$. Furthermore, the physical embodiment of the teacher is different from that of the robot and his actions (A_T) are therefore *not* the same as those of the robot (A_L). Therefore in order to make the demonstration data meaningful for the robot, a mapping $D \rightarrow A_L$ must be applied to convert the demonstration into the robot's frame of reference. This is one example of a $g_E(z, a) \neq I(z, a)$ *embodiment* mapping.

The categorization of LfD data source that we present in this article groups approaches according to the absence or presence of the record and embodiment mappings. We select this categorization to highlight the levels at which correspondence plays a role in demonstration learning. Within a given learning approach, the inclusion of each additional mapping introduces a potential injection point for correspondence difficulties; in short, the more mappings, the more difficult it is to recognize and reproduce the teacher's behavior. However, mappings also reduce constraints on the teacher and increase the generality of the demonstration technique.

In our categorization, we first split LfD data acquisition approaches into two categories based on the embodiment mapping, and thus by execution platform:

- *Demonstration*: There is no embodiment mapping, because demonstration is performed on the actual robot learner (or a physically identical platform). Thus $g_E(z, a) \equiv I(z, a)$.
- *Imitation*: There exists an embodiment mapping, because demonstration is performed on a platform which is *not* the robot learner (or a not physically identical platform). Thus $g_E(z, a) \neq I(z, a)$.

We then further distinguish approaches within each of these categories according to record mapping, relating to how the demonstration is recorded. Figure 5 introduces our full categorization of the various approaches for building the demonstration dataset. Our discussion of data acquisition in subsequent sections is structured according to this categorization.

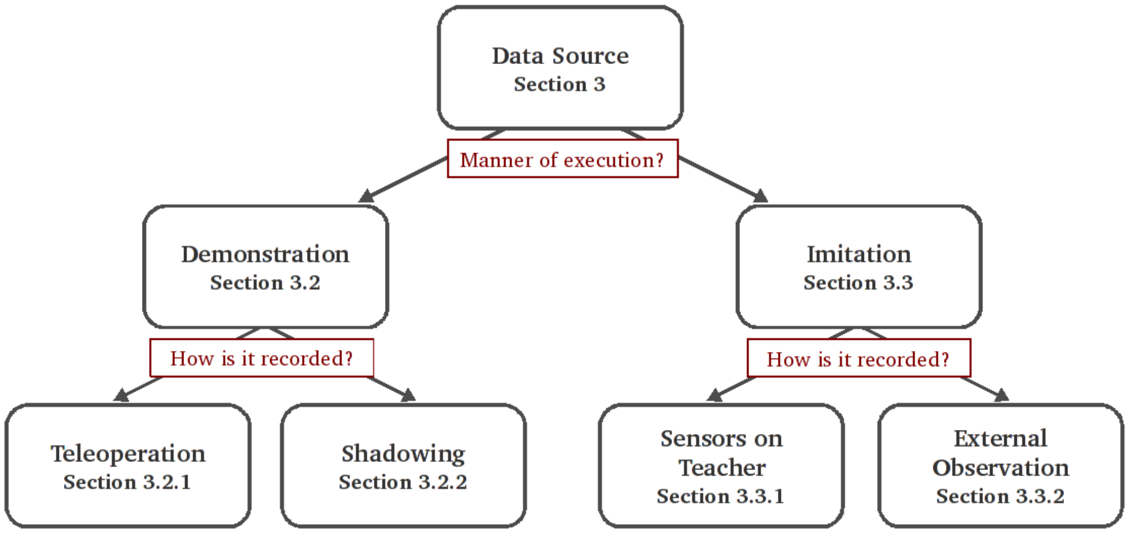

Figure 5: Categorization of approaches to building the demonstration dataset.

3.2 Demonstration

When teacher executions are *demonstrated*, by our definition there exists no embodiment mapping issue between the teacher and learner. This situation is presented within the left hand column of Figure 4. There may exist a non-direct record mapping, however, for state and/or actions. This occurs if the states experienced (actions taken) by the demonstrator are not recorded directly, and must instead be inferred from the data. Based on this distinction, we identify two common approaches for providing demonstration data to the robot learner as:

- *Teleoperation* (Sec. 3.2.1) : A demonstration technique in which the robot learner platform is *operated* by the teacher, and the robot's sensors record the execution. There record mapping is direct; thus $g_R(z,a) \equiv I(z,a)$.
- *Shadowing* (Sec. 3.2.2) : A demonstration technique in which the robot learner records the execution using its own sensors while attempting to match or *mimic* the teacher motion as the teacher executes the task. There exists a non-direct record mapping; thus $g_R(z,a) \neq I(z,a)$.

Again, for both teleoperation and shadowing the robot records from its own sensors as its body executes the behavior, and so the embodiment mapping is direct, $g_E(z,a) \equiv I(z,a)$.

The record mapping distinction plays an important role in the application and development of demonstration algorithms. As described below, teleoperation is not suitable for all learning platforms, while shadowing techniques require an additional processing component to enable the learner to mimic the teacher. In the following subsections we discuss various works that utilize these demonstration techniques.

3.2.1 Teleoperation

During teleoperation, a robot is operated by the teacher while recording from its own sensors. Since the states/actions experienced during the execution are directly recorded, the record mapping is direct and $g_R(z,a) \equiv I(z,a)$. Teleoperation provides the most direct method for information transfer within demonstration learning. However, teleoperation requires that operating the robot be manageable, and as a result not all systems are suitable for this technique. For example low level motion demonstrations are difficult on systems with complex motor control, such as high degree of freedom humanoids.

Demonstrations recorded through human teleoperation via a joystick are used in a variety of applications, including flying a robotic helicopter [Ng et al., 2004], robot kicking motions [Browning et al., 2004], object grasping [Pook and Ballard, 1993, Sweeney and Grupen, 2007], robotic arm assembly tasks [Chen and Zelinsky, 2003] and obstacle avoidance and navigation [Inamura et al., 1999, Smart, 2002]. Teleoperation is also applied to a wide variety of simulated domains, ranging from static mazes [Clouse, 1996, Rao et al., 2004] to dynamic driving [Abbeel and Ng, 2004, Chernova and Veloso, 2008b] and soccer domains [Aler et al., 2005], and many other applications.

Human teachers also employ techniques other than direct joysticking. In kinesthetic teaching, a humanoid robot is not actively controlled but rather its passive joints are moved through desired motions [Billard et al., 2006]. Demonstration may also be performed through speech dialog, in which the robot is told specifically what actions to execute in various states [Breazeal et al., 2006, Lauria et al., 2002, Rybski et al., 2007]. Both of these techniques might be viewed as variants on traditional teleoperation, or alternatively as a sort of high level teleoperation. In place of a human teacher, hand-written controllers are also used to teleoperate robots [Argall et al., 2007, Grollman and Jenkins, 2007, Rosenstein and Barto, 2004, Smart, 2002].

We pause here to note that demonstration data recorded by real robots frequently does not represent the *full* observation state of the teacher. This occurs if, while executing, the teacher employs extra sensors which are not recorded. For example, if the teacher observes parts of the world which are inaccessible from the robot's cameras (e.g. behind the robot, if its cameras are forward-facing), then state, as observed by the teacher, differs from what is actually recorded as data. A small number of works do address this problem. For example, in Grudic and Lawrence [1996] a vision-based robot is teleoperated while the teacher looks exclusively at a screen displaying the robot's camera output.

3.2.2 Shadowing

During shadowing, the robot platform mimics the teacher's demonstrated motions while recording from its own sensors. The record mapping is not direct, $g_R(z,a) \neq I(z,a)$, because the states/actions of the true demonstration execution are not recorded. Rather, the learner records its own mimicking execution, and so the teacher's states/actions are *indirectly* encoded within the dataset. In comparison to teleoperation, shadowing requires an extra algorithmic component which enables the robot to track and actively shadow (rather than passively be teleoperated by) the teacher.

Navigational tasks learned from shadowing have a robot follow an identical-platform robot teacher through a maze [Demiris and Hayes, 2002], follow a human teacher past sequences of colored markers [Nicolescu and Mataric, 2001b] and mimic routes determined from observations of human teacher executions [Nehmzow et al., 2007]. A humanoid learns arm gestures, as well as the rules of a turn-taking gesture game, by mimicking the motions of a human demonstrator [Ogino et al., 2006].

3.3 Imitation

As previously defined, embodiment issues do exist between the teacher and learner for *imitation* approaches. This situation is presented within the right hand column of Figure 4, indicating the presence of a non-identity embodiment mapping $g_E(z,a) \neq I(z,a)$. Within this setting, we further divide approaches for providing imitation data by whether the record mapping is an identity or not:

- *Sensors on Teacher* (Sec. 3.3.1) : An imitation technique in which sensors located on the executing body are used to record the teacher execution. The record mapping is direct; thus $g_R(z,a) \equiv I(z,a)$.
- *External Observation* (Sec. 3.3.2) : An imitation technique in which sensors external to the executing body are used to record the execution. These sensors may or may not be located on the robot learner. There exists a non-direct record mapping; thus $g_R(z,a) \neq I(z,a)$.

The record mapping distinction again plays an important role in the application and development of imitation algorithms, just as it did with demonstration approaches. The sensors on teacher approach provides precise measurements, but requires a lot of overhead in the way of specialized sensors and customized surroundings. By contrast, external observation is a much more general method, but also provides less reliable measurements. In the following subsections, we discuss these imitation techniques and the various works that utilize them. Additionally, we point the reader to a review by Breazeal and Scassellati [2002], which examines the problems of correspondence and knowing what to imitate.

3.3.1 Sensors on Teacher

The sensors on teacher approach utilizes recording sensors located *directly on* the executing platform. This means no record mapping, and so $g_R(z,a) \equiv I(z,a)$, which alleviates one potential source for correspondence difficulties. The strength of this technique is that precise measurements of the example execution are provided. However, the overhead attached to the specialized sensors, such as human-wearable sensor-suits, or customized surroundings, such as rooms outfitted with cameras, is non-trivial and limits the settings within which this technique is applicable.

Human teachers commonly use their own bodies to perform example executions by wearing sensors able to record the person's state and actions. This is especially true when working with humanoid or anthropomorphic robots, since the body of the robot resembles that of a human. The recorded joint angles of a human teach drumming patterns to a 30-DoF humanoid [Ijspeert et al., 2002a], and in later work walking patterns as well [Nakanishi et al., 2004]. A humanoid learns referee signals by pairing kinesthetic teachings (3.2.1) with human teacher executions recorded via wearable motion sensors [Calinon and Billard, 2007a]. Another approach has a human wearing sensors control a simulated human, which maps to a simulated robot and then to a real robot arm [Aleotti and Caselli, 2006].

3.3.2 External Observation

Imitation performed through external observation relies on data recorded by sensors located *externally to* the executing platform. This means that a record mapping exists, and $g_R(z, a) \neq I(z, a)$. Since the actual states/actions experienced by the teacher during the execution are not directly recorded, they must be inferred. This introduces a new source of uncertainty for the learner. Some LfD implementations first extract the teacher states/actions from this recorded data, and then map the extracted states/actions to the learner. Others map the recorded data directly to the learner, without ever explicitly extracting the states/actions of the teacher. Compared to the sensors on teacher approach, the data recorded under this technique is less precise and less reliable. The method, however, is more general and is not limited by the overhead of specialized sensors and settings.

Early work by Atkeson and Schaal [1997] has a robotic arm learn pole balancing via stereo-vision and human demonstration. Unlike most other approaches in this section, here human demonstration trains a non-anthropomorphic robot and data is obtained by tracking the movement of the pole, not the teacher's arm itself. Similarly, in Bentivegna et al. [2002] the position of the human player's (teacher's) puck is tracked when teaching a 37-DoF humanoid to play air hockey. Here the robot learner itself directly observes the teacher executions. This is representative of other humanoid systems, where cameras are often located directly on the robot body as a parallel to human eyes.

Typically, the external sensors used to record human teacher executions are vision-based. Motion capture systems utilizing visual markers are applied to teaching human motion [Amit and Mataric, 2002, Billard and Mataric, 2001, Ude et al., 2004] and manipulation tasks [Pollard and Hodgins, 2002]. Visual features are tracked in biologically-inspired frameworks that link perception to abstract knowledge representation [Chella et al., 2006] and teach an anthropomorphic hand to play the game Rock-Paper-Scissors [Infantino et al., 2004]. Background subtraction is used by Ogino et al. [2006] to extract teacher motion from images. These authors take an unique approach to building the embodiment mapping $g_E(z, a)$; the correspondence is *learned*, rather than being hand-engineered, by having the human teacher shadow the robot's motions.

A number of systems combine external sensors with other information sources; in particular, with sensors located directly on the teacher. A force-sensing glove is combined with vision-based motion tracking to teach grasping movements [Lopes and Santos-Victor, 2005, Voyles and Khosla, 2001]. Movement, end effector position, stereo vision, and tactile information teaches generalized dexterous motor skills to a humanoid robot [Lieberman and Breazeal, 2004]. The combination of 3D marker data with torso movement and joint angle data is used for applications on a variety of simulated and robot humanoid platforms [Mataric, 2002]. Other approaches combine speech with visual observation of human gestures [Steil et al., 2004] and object tracking [Demiris and Khadhouri, 2006], within the larger goal of speech-supported LfD learning.

Several works also explore learning through external observation of non-human teachers. A robot learns a manipulation task through visual observation of an identical robotic teacher [Dillmann et al., 1995], and a simulated agent learns about its own capabilities and unvisited parts of the state space through observation of other simulated agents [Price and Boutilier, 2003]. A generic framework for solving the correspondence problem between differently embodied robots is presented in Alissandrakis et al. [2004], where a robotic agent learns new behaviors through imitation of another, possibly physically different, agent.

3.4 Other Approaches

Within LfD there do exist exceptions to the data source categorization we have presented here. These exceptions record only states during demonstration, without recording actions. For example, by drawing a path through a 2-D representation of the physical world, high level path-planning demonstrations are provided to a rugged outdoor robot [Ratliff et al., 2006] and a small quadruped robot [Ratliff et al., 2007, Kolter et al., 2008]. Since actions are not provided in the dataset, no state-action mapping is learned for action selection. Instead, actions are selected at runtime by employing low level motion planners and controllers [Ratliff et al., 2006, 2007], or by providing transition models $T(s'|s, a)$ [Kolter et al., 2008].

4 Deriving a Policy: The Source of the State to Action Mapping

Given a dataset of state-action examples that have been acquired using one of the methods described in the previous section, we now discuss methods for deriving a policy using this data. LfD has seen the development of three core approaches to deriving policies from demonstration data, as summarized in Figure 2. Learning a policy can involve simply learning an approximation to this mapping (*mapping function*), or learning a model of the world dynamics and deriving a policy from this information (*system model*). Alternately, a sequence of actions can be produced by a planner after learning a model of action pre- and post-conditions (*plans*). Across all of these learning techniques, minimal parameter tuning and fast learning times requiring few training examples are desirable.

We introduce a full categorization of the various approaches to deriving a policy from the demonstration dataset in Figure 6. Approaches are initially split between the three core derivation techniques described above. Further splits, if present, are approach-specific. We discuss each of these categories in depth in the following sections. Additionally, we direct the reader to a literature review by Schaal et al. [2003] that focuses on statistical and mathematical approaches to deriving motor control policies.

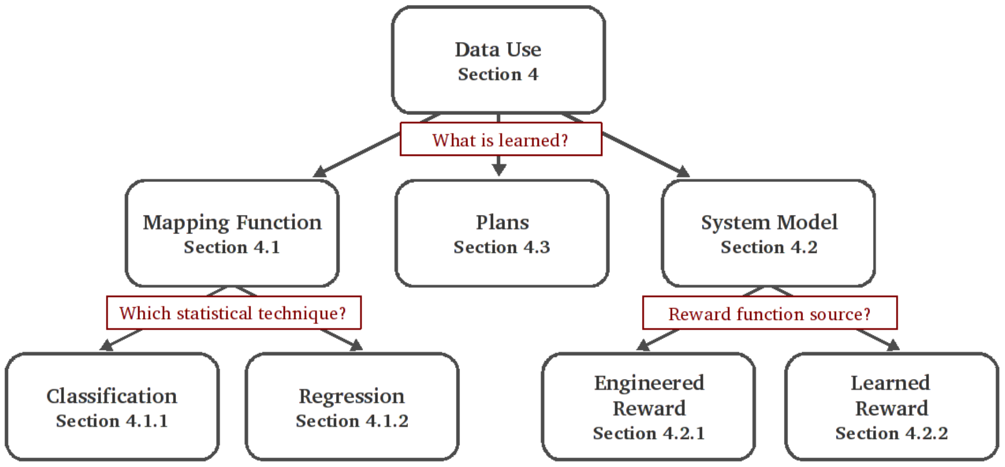

Figure 6: Categorization of approaches to learning a policy from demonstration data.

4.1 Mapping Function

The *mapping function* approach to policy learning calculates a function that approximates the state to action mapping, $f() : Z \rightarrow A$, for the demonstrated behavior (Fig. 2a). The goal of this type of algorithm is to reproduce the underlying teacher policy, which is unknown, and to generalize over the set of available training examples such that valid solutions are also acquired for similar states that may not have been encountered during demonstration.

The details of function approximation are influenced by many factors. These include whether the state input and action output are continuous or discrete, whether the generalization technique uses the data to approximate a function prior to execution time or directly at execution time, whether it is feasible or desirable to keep the entire demonstration dataset around throughout learning, and whether the algorithm updates online.

In general, mapping approximation techniques fall into two categories depending on whether the prediction output of the algorithm is discrete or continuous. *Classification* techniques are used to produce discrete output (Sec. 4.1.1), and *regression* techniques produce continuous output (Sec. 4.1.2). Many techniques for performing classification and regression have been developed outside of LfD, and we refer the reader to Hastie et al. [2001] for a full discussion.

4.1.1 Classification

Classification approaches categorize their input into discrete classes, thereby grouping similar input values together. In the context of policy learning, the input to the classifier is robot states and the discrete output classes are robot actions. Below, we present a summary of classification methods applied at three action control levels (basic motion control, action primitives, complex behaviors), as defined in Section 2.2. Although a single classification algorithm can be applied at any level, we distinguish between them to highlight the generality of this learning technique.

Low level robot actions include basic commands such as moving forward or turning. Example applications that learn a mapping from states to low level actions include controlling a car within a simulated driving domain using *Gaussian Mixture Models (GMMs)* [Chernova and Veloso, 2007], flying a simulated airplane using *decision trees* [Sammut et al., 1992] and learning obstacle avoidance and navigation behaviors using *Bayesian network* [Inamura et al., 1999] and *k-Nearest Neighbors (kNN)* [Saunders et al., 2006] classifiers.

When states are mapped to motion primitives, the primitives typically are then composed or sequenced together. For example, Pook and Ballard [1993] classify primitive membership using kNN, and then recognize each primitive within the larger demonstrated task via *Hidden Markov Models (HMMs)*, to teach a robotic hand and arm an egg flipping manipulation task. HMMs are also used to teach a basic assembly task [Hovland et al., 1996], and to learn motor-skill tasks by identifying and generalizing upon the intentions of the user [Dixon and Khosla, 2004]. A biologically-inspired framework is presented in Mataric [2002], which automatically extracts primitives from demonstration data, classifies data via *vector-quantization* and then composes and/or sequences primitives within a hierarchical Neural Network. The work is applied to a variety of test beds, including a 20 DoF simulated humanoid torso, a 37 DoF avatar (a simulated humanoid which responds to external sensors), Sony AIBO dogs and small differential drive Pioneer robots. Under this framework, the simulated humanoid torso is taught a variety of dance, aerobics and athletics motions [Jenkins et al., 2000], the avatar reaching patterns [Billard and Mataric, 2001] and the Pioneer sequenced location visiting tasks [Nicolescu and Mataric, 2001a].

Similar approaches have been used for the classification of high-level behaviors. The behaviors themselves are generally developed (by hand or learned) prior to task learning. Gesture behaviors for an anthropomorphic hand are recognized through a similarity measure on an eigenvector representation [Voyles and Khosla, 2001], and within this framework HMMs classify demonstrations into gestures for a box sorting task with a Pioneer robot [Rybski and Voyles, 1999]. The Bayesian likelihood method is used to select actions for a humanoid robot in a button pressing task [Lockerd and Breazeal, 2004], and *Support Vector Machines (SVMs)* classify behaviors for a robotic ball sorting task [Chernova and Veloso, 2008a].

4.1.2 Regression

Regression approaches map demonstration states to continuous action spaces. Similar to classification, the input to the regressor is robot states, and the *continuous* output are robot actions. Since the continuous-valued output often results from combining multiple demonstration set actions, typically regression approaches are applied to low-level motions and not high-level behaviors. A key distinction between methods is whether the mapping function approximation occurs at run time, or prior to run time. Below we present a summary of regression techniques for LfD along a continuum between these two extremes.

At one extreme lies Lazy Learning [Atkeson et al., 1997], where function approximation does not occur until a current observation point in need of mapping is present. The simplest Lazy Learning technique is kNN, which is applied to action selection within robotic marble-maze [Bentivegna, 2004] and simulated ball interception [Argall et al., 2007] domains. More complex approaches include *Locally Weighted Regression (LWR)* [Cleveland and Loader, 1995]. One LWR technique further anchors local functions to the phase of nonlinear oscillators [Schaal and Sternad, 1998] to produce rhythmic movements, specifically drumming [Ijspeert et al., 2002b] and walking [Nakanishi et al., 2004] patterns with a humanoid robot. While Lazy Learning approaches are fast and expend no effort approximating the function in areas which are unvisited during execution time, they do require keeping around all of the training data.

In the middle of the data processing continuum lie techniques in which the original data is converted to another, possibly sparsified, representation prior to run time. This *converted* data is then used by Lazy Learning techniques at run time. For example, *Receptive Field Weighted Regression (RFWR)* [Schaal and Atkeson, 1998] first converts demonstration data to a Gaussian and local linear model representation. *Locally Weighted Projection Regression (LWPR)* [Vijayakumar and Schaal, 2000] extends this approach to scale with input data dimensionality and redundancy. Both RFWR and LWPR are able to incrementally update the number of representative Gaussians as well as regression parameters online. Successful robotics applications using LWPR include an AIBO robot performing basic soccer skills [Grollman and Jenkins, 2007] and a humanoid playing the game of air hockey [Bentivegna, 2004]. The work of Bentivegna actually employs multiple techniques, first using kNN to select a behavior class and then LWPR to generate low-level actions using the demonstration data within this class. The approaches at this position on the continuum benefit from not needing to evaluate all of the training data at run time, but at the cost of extra computation and generalization prior to execution.

At the opposite extreme lie approaches which form a complete function approximation prior to execution time. At run time, they no longer depend on the presence of the underlying data (or any modified representations thereof). A seminal work by Pomerleau [1991] uses a *Neural Network (NN)* to enable autonomous driving of a van at speed on a variety of road types. NN techniques also enable a robot arm to perform a peg in hole task [Dillmann et al., 1995], and a set of b-spline wavelets approximate time-sequenced data for full body humanoid motion [Ude et al., 2004]. Also possible are statistical approaches that represent the demonstration data in a single or mixture distribution which is sampled at run time. Demonstration data is encoded as a joint distribution over objects, hand position and hand orientation, and is used by a humanoid for grasping novel and known household objects [Steil et al., 2004]. A humanoid robot is taught basketball referee signals using *Gaussian Mixture Regression (GMR)* [Calinon and Billard, 2007a], and an AIBO robot to perform basic soccer skills using *Sparse On-Line Gaussian Processes (SOGP)* [Grollman and Jenkins, 2008]. The regression techniques in this area all have the advantage of *no* training data evaluations at run time, but are the most computationally expensive prior to execution. Additionally, some of the techniques, for example NN, can suffer from "forgetting" mappings learned earlier in the learning run, a problem of particular importance when updating the policy online.

4.2 System Models

The *system model* approach to LfD policy learning uses a state transition model of the world, $T(s'|s,a)$, from which a policy $\pi : Z \rightarrow A$ is then derived (Fig. 2b). This approach is typically formulated within the structure of *Reinforcement Learning (RL).* The transition function $T(s'|s,a)$ generally is determined based on demonstration data and any additional autonomous exploration the robot may do. To derive a policy from this transition model, a reward function $R(s)$, which associates reward value r with world state s, must be either learned from demonstrations or defined by the user.

The goal of RL is to maximize cumulative reward over time. The expected cumulative future reward of the state under the current policy is represented by further associating each state s with a value according to the function $V(s)$ (or associating state-action pair s, a with a Q-value according to the function $Q(s,a)$). State values may be represented by the *Bellman Equation*:

$$V^{\pi}(s) \;=\; \int_a \pi(s,a) \int_{s'} T(s'|s,a)\,[r(s) + \gamma V^{\pi}(s')] \qquad (1)$$

where $V^{\pi}(s)$ is the value of state s under policy π, and γ represents a discounting factor on future rewards. How to use these values to update a policy, and how to update them effectively online, is a subject of much research outside of LfD. Unlike function approximation techniques, RL approaches typically do *not* generalize state and demonstrations must be provided for every discrete state. For a full review of RL, we refer the reader to Sutton and Barto [1998].

Reward is the motivation behind state and action selection, and consequently also guides task execution. Defining a reward function to accurately capture the desired task behavior, however, is not always obvious. We therefore categorize LfD implementations of the system model method by the source of the reward function. Some take the approach of an *engineered* reward function, as seen in classical RL implementations (Sec. 4.2.1). In other approaches, the reward function is *learned* from the demonstration data (Sec. 4.2.2).

4.2.1 Engineered Reward Functions

Within most applications of the LfD system model approach, the reward function is manually defined by the user. User-defined rewards tend to be sparse, meaning that the reward value is zero except for a few states, such as around obstacles or near the goal. Various demonstration-based techniques therefore have been defined to aid the robot in locating the rewards and to prevent extensive periods of blind exploration.

Demonstration may be used to highlight interesting areas of the state space in domains with sparse rewards, for example using teleoperation to show the robot the reward states and thus eliminating long periods of initial exploration in which no reward feedback is acquired [Smart and Kaelbling, 2002]. Both reward and action demonstrations influence the performance of the learner in the supervised actor-critic reinforcement learning algorithm [Sutton and Barto, 1998]. Applied to a basic assembly task using a robotic manipulator [Rosenstein and Barto, 2004], reward feedback penalizes negative and reinforces positive behavior, and action demonstrations highlight recommended actions and suggest promising directions for state exploration.

A number of algorithms that use hand-engineered rewards also use interactions with other autonomous agents to enhance learning. However, these approaches have mostly been studied in simulation. In the Ask For Help framework [Clouse, 1996], RL agents request advice from other similar agents when all possible actions in a given state have relatively equal quality estimates. In a similarly motivated system [Oliveira and Nunes, 2004], agents are able to select, exchange and incorporate advice from other agents, and to combine this with information acquired locally through exploration, to speed up learning. Novice agents learn in a multiagent system [Price and Boutilier, 2003] by passively observing expert agents in the environment, without explicit teaching.

Finally, real robot applications tend to employ RL in ways not typically seen in the classical RL policy derivation algorithms. This is because even in discrete state-action spaces the cost of visiting every state, and taking every action from each state, becomes prohibitively high and further could be physically dangerous. One approach is to use simulation to seed an initial world model, for example using a planner operating on a simulated version of a marble maze robot [Stolle and Atkeson, 2007]. The opposite approach derives a model of robot dynamics from demonstration but then uses the model in simulation, for example to simulate robot state when employing RL to optimize a NN controller for autonomous helicopter flight [Bagnell and Schneider, 2001] and inverted helicopter hovering [Ng et al., 2004]. The former implementation additionally pairs the hand-engineered reward with a high penalty for visiting any state not encountered during demonstration, and the latter additionally makes the explicit assumption of a limited class of feasible controllers.

4.2.2 Learned Reward Functions

Defining an effective reward function in real world systems can be a non-trivial issue. One subfield within RL that addresses this is *Inverse Reinforcement Learning* [Russell, 1998], where the reward function is *learned* rather than hand-defined. LfD has seen the development of several algorithms, within a variety of system model approaches, that examine learning a reward function from demonstrations.

A reward function is learned by associating greater reward with states similar to those encountered during demonstration, and is combined with RL techniques to teach a pendulum swing-up task to a robotic arm [Atkeson and Schaal, 1997]. In a modified RL algorithm [Thomaz and Breazeal, 2006] the task reward is learned entirely based on individual rewards interactively provided by a human user. An additional guidance component also enables the user to recommend actions for the robot to perform.

Rewarding similarity between teacher demonstrated and robot-executed trajectories is another approach to learning a reward function. Example approaches include a Bayesian model formulation [Ollis et al., 2007], and another that further provides greater reward to positions close to the goal and thus enables adaptation to task changes like a new goal location or path obstacles [Guenter and Billard, 2007]. A trajectory comparison based on state features first pioneered in Abbeel and Ng [2004] guarantees that a correct *reward* function is learned if the feature counts are properly estimated. This guarantee is extended to learning the correct *behavior*, and furthermore is implemented on a real robot system, in Ratliff et al. [2006]. Here a mapping from observation input to state costs is learned using margin-maximization techniques, and the resulting terrain cost map is then used by a navigational planner to generate goal-achieving paths as sequences of states. This work is extended to consider the decomposition of task demonstration into hierarchies for a small legged robot [Kolter et al., 2008, Ratliff et al., 2007], and further to resolve ambiguities in feature counts and reward functions by employing the principle of maximum entropy [Ziebart et al., 2008]. All of these works learn the reward function $R(s)$ exclusively; however, both a reward function *and* transition function $T(s'|s,a)$ are learned for acrobatic helicopter maneuvers in Abbeel et al. [2007].

4.3 Plans

An alternative to mapping states directly to actions is to represent the desired robot behavior as a plan (Fig. 2c). Within the planning framework, the policy is represented as a sequence of actions that lead from the initial state to the final goal state. Actions are often defined in terms of *pre-conditions*, the state that must be established before the action can be performed, and *post-conditions*, the state resulting from the action's execution. Unlike other LfD approaches, planning techniques frequently rely not only on state-action demonstrations, but also on additional information in the form of *annotations* or *intentions* from the teacher. Demonstration-based algorithms differ in how the rules associating pre- and post-conditions with actions are learned, and whether additional information is provided by the teacher.

One of the first papers to learn execution plans based on demonstration is Kuniyoshi et al. [1994], in which a plan is learned for object manipulation based on observations of the teacher's hand movements. A generalized plan, able to represent sequential tasks with repetitions, is learned from only two demonstrations by a humanoid performing a repetitive ball collection task [Veeraraghavan and Veloso, 2008]. Other approaches use spoken dialog both as a technique to demonstrate task plans, and also to enable the robot to verify unspecified or unsound parts of a plan through dialog. Under such a framework, multiple teachers provide verbal navigation instructions, with the intention of building a vocabulary that maps to sensorimotor primitives [Lauria et al., 2002], and a single teacher presents a robot with a series of conditional statements that are then processed into a plan [Rybski et al., 2007].

Other planning-based methods require teacher annotations. Overall task intentions provided in response to learner queries are used to encode plan post-conditions [Friedrich and Dillmann, 1995], and human feedback draws attention to particular elements of the domain when learning a high level plan from shadowing [Nicolescu and Mataric, 2003]. Goal annotations include indicating the demonstrator's goal and time points at which it is achieved or abandoned when learning non-deterministic plans [van Lent and Laird, 2001], and providing binary feedback that grows or prunes the goal set used to infer the demonstrator's goal [Jansen and Belpaeme, 2006]. Annotations of the demonstrated action sequence are combined with high level instructions, for example that some actions can occur in any order, when learning a domain-specific hierarchical task model [Garland and Lesh, 2003].

5 Limitations of the Demonstration Dataset

LfD systems are inherently linked to the information provided in the demonstration dataset. As a result, learner performance is heavily limited by the quality of this information. In this article we identify two distinct causes for poor learner performance within LfD frameworks and survey the techniques that have been developed to address each limitation. The first cause, discussed in Section 5.1, is due to dataset sparsity, or the existence of areas of the state space that have not been demonstrated. The second, discussed in Section 5.2, is due to poor quality of the dataset examples, which can result from a teacher's inability to perform the task optimally.

5.1 Undemonstrated State

All LfD policies are limited by the availability of training data because, in all but the most simple domains, the teacher is unable to demonstrate the correct action from every possible state. This raises the question of how the robot should act when it encounters a state for which no demonstration has been provided. We categorize existing methods for dealing with undemonstrated state into two approaches: *generalization from existing demonstrations*, and *acquisition of new demonstrations.*

5.1.1 Generalization from Existing Demonstrations

Here we discuss methods for using the existing data to deal with undemonstrated state. We present techniques as used within each of the core approaches for policy derivation.

Within the function mapping approach, undemonstrated state is dealt with through state generalization. Generalizing across inputs is a feature inherent to robust function approximation. The exact nature of this generalization depends upon the specific classification or regression technique used. Approaches range from strict grouping with the nearest dataset point state, as in 1-Nearest Neighbor, to soft averaging across multiple states, as in kernelized regression.

Within the system model approach, undemonstrated state can be addressed through state exploration. State exploration is often accomplished by providing the learner with an exploration policy that guides action selection within undemonstrated states. Based on rewards provided by the world, the performances of these actions are automatically evaluated. This motivates the issue of *Exploration vs. Exploitation*; that is, of determining how frequently to take exploratory actions versus following the set policy. We note that taking exploratory steps on a real robot system is often unwise, for safety and stability reasons. A safe exploration policy is employed by Smart [2002] for the purposes of gathering robot demonstration data, both to seed and then update a transition model and reward function. To guide exploration within this work, initially a suboptimal controller is provided, and once a policy is learned small amounts of Gaussian noise are randomly added to the greedy actions of this policy.

Generalization to unseen states is not a common feature amongst traditional planning algorithms. This is in large part due to the common assumption that every action has a set of known, deterministic effects on the environment that lead to a particular known state. However, this assumption is typically dropped in real-world applications, and among LfD approaches several algorithms do explore the generalization of demonstrated sequences. For example, Friedrich and Dillmann [1995] present a generalization method in which the teacher is able to specify the level of generality that was implied by a demonstration, by answering the learner's questions (e.g. should the box be picked up only from the chair, or from any surface besides the table?). Other approaches focus on the development of generalized plans with flexible action order [van Lent and Laird, 2001, Nicolescu and Mataric, 2003].

5.1.2 Acquisition of New Demonstrations

A fundamentally different approach to dealing with undemonstrated state is to re-engage the teacher and acquire additional demonstrations when novel states are encountered by the robot. Under such a paradigm, the responsibility of selecting states for demonstration now is shared between the robot and teacher.

One set of approaches acquire new demonstrations by enabling the learner to evaluate its confidence in selecting a particular action, based on the confidence of the underlying classification algorithm. In Chernova and Veloso [2007, 2008b] the robot requests additional demonstration in states that are either very different from previously demonstrated states or states in which a single action can not be selected with certainty. In Grollman and Jenkins [2007] the robot indicates to the teacher its certainty in performing various elements of the task, and the teacher may choose to provide additional demonstrations indicating the correct behavior to perform.

5.2 Poor Quality Data

The quality of a learned LfD policy depends heavily on the quality of the provided demonstrations. In general, approaches assume the dataset to contain high quality demonstrations performed by an expert. In reality, however, teacher demonstrations may be ambiguous, unsuccessful or suboptimal in certain areas of the state space. A naively learned policy will likely perform poorly in such areas, as shown in the seminal work by Atkeson and Schaal [1997]. In this section we discuss proposed approaches for improving demonstration data and dealing with poor learner performance.

5.2.1 Suboptimal or Ambiguous Demonstrations

Inefficient demonstrations may be dealt with by eliminating parts of the teacher's execution that are suboptimal. Other approaches address the ambiguities that can result from single or multiple demonstrations.

Suboptimality is most common in demonstrations of low level tasks, such as movement trajectories for robotic arms. In these domains, the demonstration serves as guidance instead of offering a complete solution. Kaiser et al. [1995] present a method for actively identifying and removing elements of the demonstration that are unnecessary or inefficient, by removing actions that do not contribute to the solution to the task. Other approaches smooth or generalize over suboptimal demonstrations in such a way as to improve upon the teacher's performance. For example, when learning motion control, data from multiple repeated demonstrations is used to obtain a smoothed optimal path [Aleotti and Caselli, 2006, Delson and West, 1996, Yeasin and Chaudhuri, 2000], and data from multiple teachers encourages more robust generalization [Pook and Ballard, 1993].

Dataset ambiguity may be caused by uncaptured elements of the record mapping. Differences in teacher and learner perspectives during externally recorded demonstrations are accounted for in Breazeal et al. [2006], by having the robot maintain separate belief estimates for itself and the human demonstrator. Additionally, dataset ambiguity may be caused by teacher demonstrations that are inconsistent between multiple executions. This type of ambiguity can arise due to the presence of multiple equally applicable actions (e.g. both turning left and moving forward are equally valid actions) among which the teacher selects arbitrarily, or due to state corruption through sensor noise. The result in both cases is that identical, or nearly identical, states are mapped to different actions in the demonstration. Approaches designed to address these inconsistencies include requests for additional clarification demonstrations from the teacher [Chernova and Veloso, 2007] and modeling the choice of actions explicitly in the robot policy [Chernova and Veloso, 2008c].

5.2.2 Learning from Experience

An all together different approach to dealing with poor quality data is for the student to learn from experience. If the learner is provided with feedback that evaluates its performance, this may be used to update its policy. This evaluation is generally provided via teacher feedback, or through a reward as in RL.

Automatic policy evaluations are employed in Bentivegna [2004], where a policy *derived* under the mapping function approximation approach is *updated* using RL techniques. The function approximation takes into consideration Q-values associated with paired query-demonstration points, and these Q-values are updated based on learner executions and a developer-defined reward function. A student uses binary policy evaluations from a human teacher in Jansen and Belpaeme [2006] to prune the set of inferred teacher goals that it maintains. A human-provided binary tag flags portions of the learner execution for poor performance in Argall et al. [2007], and the learner uses this to update a continuous-action mapping function approximation.

Most use of RL-type rewards, however, occurs under the system model approach. We emphasize that these are rewards seen during *learner* execution with its policy, and not during teacher demonstration executions. Smart [2002] seeds the robot's reward and transition functions with a small number of suboptimal demonstrations, and then optimizes the policy through exploration and RL. A similar approach is taken by Peters and Schaal [2008] in their use of the Natural Actor Critic algorithm, a variant of RL, to learn a motor primitives for a 7-DoF robotic arm. In Stolle and Atkeson [2007], the world model is seeded with trajectories produced by a planner operating on a simulated version of their robot, and learner execution results in state values being updated, and consequently also the policy. In addition to updating state values, student execution information may be used to update the actual learned dynamics model $T(s'|s,a)$. This approach is taken by Abbeel and Ng [2005], who seed their world model with teacher demonstrations. Policy updates then consist of rederiving the world dynamics after adding learner executions with this policy (and later updated policies) to the demonstration set.

Beyond providing just an evaluation of performance, teacher feedback may also provide a *correction* on the executed behavior. The correct action from a discrete set is provided by a human teacher in Nicolescu and Mataric [2003], and this information updates the structure of a hierarchical Neural Network of robot behaviors. Discrete action corrections from a human teacher update an action classifier in Chernova and Veloso [2008c]. Further challenging is for a human to provide corrections in continuous action spaces. Kinesthetic teaching, where passive humanoid joints are moved by the teacher through the desired motions, provides continuous-valued corrections in Calinon and Billard [2007a]. A set of mathematical operations transform high level advice from a human teacher into continuous-valued corrections on students executions in Argall et al. [2008].

Before concluding, we underline the advantage of providing demonstrations and employing LfD *before* using traditional policy learning techniques, such as exploration-based methods like RL. As mentioned in Section 4.2.1, many traditional learning approaches are not immediately applicable to robots. This is largely due to the constraints of gathering experience on a real system. To physically execute *all* actions from *every* state will likely be infeasible, may be dangerous, and will not scale with continuous state spaces. However, the use of these techniques to evaluate and improve upon the experiences gained through an LfD system has proven very successful. We consider this to be an area with much potential for future research, which will be discussed further within Section 6.4.

6 Future Directions

As highlighted by the discussion in the previous sections, current approaches to LfD have been designed to address a wide variety of problems under many different conditions and assumptions. In this section, we aim to highlight several promising areas of LfD research that have received limited attention, ranging from data representation to issues of system robustness and evaluation metrics.

6.1 Learned State Features

Within all policy learning approaches, state information is typically represented by a set of *state features* that describe various attributes of the environment. The set of features available for learning is determined by the programmer or user before learning begins. A robot learning a task knows nothing about the world beyond the features it has been provided. In selecting features, programmers face a decision between a larger number of features, which provide a richer representation and allow a wider range of tasks to be learned, and a smaller number of features, which typically results in faster learning times with less training data.

Most policy learning algorithms are typically provided with exactly the data needed to learn a task, nothing more and nothing less. Some approaches, however, are able to deal with extra features by generalizing over irrelevant features [Grollman and Jenkins, 2007] or filtering them out through feature selection [Guyon and Elisseeff, 2003].

In addition to facing these challenges, we propose that future LfD algorithms are likely to face a new problem – *insufficient features*. Unlike standard robotics approaches, LfD does not require extensive programming experience but rather only the ability to demonstrate the task in the chosen manner. Due to the intuitive nature of demonstration, LfD algorithms have the potential to make robot programming accessible for everyday users who have no programming experience and want to customize robot behaviors. The use of current approaches by the general public, however, is likely to lead to situations in which users attempt to teach the robot tasks that cannot be learned simply due to the lack of features describing some aspect of the task. An open area of future work is to address this shortcoming by identifying intuitive methods for introducing new features into the learning system.

6.2 Temporal Data

Actions within almost all robot behaviors have a temporal ordering, a natural progression in which actions of the task are performed. Many policy learning algorithms, especially within the area of planning, naturally encode the temporal ordering. However, the vast majority of LfD algorithms developed to date result in a purely reactive policy that maps directly from a state to an action without considering past history or events. Reactive approaches have repeatedly been shown to achieve good performance, however they suffer from a number of drawbacks. By discarding all temporal information from the training data, such algorithms have a difficulty representing repetitive tasks, such as ones in which an action sequence must be repeated some fixed number of times. Additionally, learning becomes difficult in the presence of actions which have no perceivable effect on the environment and that therefore do not change the state of the robot. Both of these challenges can be solved by including special state features that represent the memory of recent events of this nature; for example, in Ijspeert et al. [2002b] demonstrations of repetitive rhythmic tasks are anchored to the phase of a nonlinear oscillator. However, such features are always task specific and do not present a general solution to this problem. We believe that a more universal solution can be achieved through the use of temporal information encoded in the demonstration sequence. The development of such generalized LfD policy learning techniques is an open area for future research.

6.3 Execution Failures

With any LfD approach, it is possible that the policy will be unable to execute as intended by the demonstration teacher. This could be, for example, because the state representation does not model crucial aspects of the world, or because the policy failed to generalize properly from the demonstration data. All LfD systems run the risk of encountering failures. The ability of a system to identify and address failures and their causes is both extremely valuable and extremely non-trivial.

A small number of works do directly address policy failures. Policy *derivation* failure is considered in Rybski and Voyles [1999], by identifying when a HMM fails to classify a demonstration and requesting that the human teacher re-demonstrate the task. Nicolescu and Mataric [2001a] consider policy *execution* failures: a robot learner recognizes when it is unable to complete its task (for example, due to a physical obstruction) and solicits the help of a human. Though the robot identifies task failure, it does not identify the failure cause or modify its policy; the intent is that human aid will enable task completion (for example, remove the obstruction).

6.4 New Techniques for Learning from Experience

The ability to learn from experience is a desirable trait in any learning system. The most popular approaches implemented within LfD systems to date offer reward-type evaluations of policy performance (Sec. 5.2.2). Reward functions are often sparse, however, and only offer an indication of how desirable being in a given state is; reward gives no indication of which action would have been more appropriate to select instead, for example. Richer forms of performance evaluation, however, could benefit approaches able to incorporate policy performance feedback.

There is no reason to expect that formalizing these richer evaluations would be any easier than the admittedly challenging task of formally defining reward functions (Sec. 4.2.2); if anything, a richer evaluation may be even more difficult to capture. We believe one promising solution is to have the demonstration teacher provide feedback on learner performance, in addition to providing the example executions. Demonstration teachers presumably posses a measure of task expertise, and so at the very least have existing measures by which to evaluate their *own* performance on the task. Furthermore, particularly if the teacher is human, measures that are difficult to formalize or parameterize but easy for a human to evaluate could prove particularly useful. Steps in this direction are taken by utilizing human teacher evaluations to provide corrections on discrete action selection [Nicolescu and Mataric, 2003, Chernova and Veloso, 2008b] and state sequencing [Kolter et al., 2008], continuous action selection [Argall et al., 2008] and to penalize poorly performing demonstration data [Argall et al., 2007]. Exactly *how to provide* effective feedback, and then *how to use* this information, however is for the most part still an area open for future research.

6.5 Multi-Robot Demonstration Learning

LfD approaches to date focus almost exclusively on the problem of a single robot being taught by a single teacher. However, solutions to complex tasks often require the cooperation of multiple robots. Multi-robot coordination has been extensively studied in robotics research and presents many challenges, such as issues of action coordination, communication, noise in shared information, and physical interaction. Current LfD research is beginning to take steps in this direction. Oliveira and Nunes [2004] present a study in which multiple autonomous agents request advice and provide demonstrations for each other. Alissandrakis et al. [2004, 2007] enables agents with both similar and dis-similar embodiments to learn movements from each other through imitation. Chernova and Veloso [2008a] present a confidence-based algorithm which enables a single person to teach multiple robots at the same time through the demonstration of physical and communication actions required to perform the task. These studies only begin to explore the various applications of LfD in the multi-robot setting, and many directions remain open for future work.

6.6 Evaluation Metrics

Learning from demonstration is a relatively young but rapidly growing field. As highlighted by this survey, a wide variety of approaches have been proposed for solving the challenges presented by this learning method. However, to date, little direct comparison between algorithms has been performed.

One reason for this is that most approaches have been tested using only a single domain and robotic platform. As a result, such techniques are often customized to that particular domain and do not present a general solution to a wide class of problems. Additionally, the field of LfD research currently lacks a standard set of evaluation metrics, further complicating performance comparisons across algorithms and domains.

We therefore believe that improved methods for evaluation and comparison are a fundamentally important area for future work. Several evaluation methods have already been proposed for specific areas of LfD [Alissandrakis et al., 2006, Calinon and Billard, 2007b, Nehaniv and Dautenhahn, 2001, Pomplun and Mataric, 2000]. Several metrics from the Human-Robot Interaction (HRI) community could also be adopted [Steinfeld et al., 2006]. The formalization of evaluation criteria would help to drive research and the development of widely applicable general-purpose learning techniques.

7 Conclusion

In this article we have presented a comprehensive survey of *Learning from Demonstration (LfD)* techniques employed to address the robotics control problem. LfD has the attractive characteristics of being an intuitive communication medium for human teachers and of opening control algorithm development to non-robotics-experts. Additionally, LfD complements many traditional policy learning techniques, offering a solution to some of the weaknesses in traditional approaches. Consequently, LfD has been successfully applied to many robotics applications.

The LfD robotics community, however, suffers from the lack of an established structure in which to organize approaches. In this survey, we have contributed such a structure, through a categorization of LfD techniques. We first segment the LfD learning problem into two fundamental phases. The first phase consists of *gathering* the demonstration examples. Within this phase we categorize according to how the demonstration is *executed* and how the demonstration is *recorded*. The second phase consists of *deriving* a policy from the demonstration examples. Within this phase we categorize according to what is learned: an approximation to the state-action mapping (*mapping function*), a model of the world dynamics and/or reward (*system model*), or a model of action pre- and post-conditions (*plans*). We have further contributed a categorization of current approaches into this structure.

Though LfD has proven a successful tool for robot policy development, there still exist many open areas for research, several of which we have identified. In particular, learning state features and considering the temporal nature of demonstration data are interesting directions to explore within the context of building a dataset and policy. System robustness could increase by explicitly dealing with policy derivation and execution failures. New approaches to learning from experience, that are specific to LfD systems, hold much promise for improving policy performance, and LfD extends naturally into multi-robot domains, which is an intriguing area that remains largely open. Finally, a standardized set of evaluation metrics would facilitate comparisons between approaches and implementations, to the benefit of the LfD community as a whole.

8 Acknowledgement

This research is partly sponsored by the Boeing Corporation under Grant No. CMU-BA-GTA-1, BBNT Solutions under subcontract No. 950008572, via prime Air Force contract No. SA-8650-06-C-7606, and the Qatar Foundation for Education, Science and Community Development. The views and conclusions contained in this document are solely those of the authors.

The authors would like to thank J. Andrew Bagnell and Darrin Bentivegna for feedback on the content and scope of this article.

References

P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine Learning (ICML'04)*, 2004.

P. Abbeel and A. Y. Ng. Exploration and apprenticeship learning in reinforcement learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML'05)*, 2005.

P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Proceedings of Advances in Neural Information Proccessing (NIPS'07)*, 2007.

J. Aleotti and S. Caselli. Robust trajectory learning and approximation for robot programming by demonstration. *Robotics and Autonomous Systems, Special Issue on The Social Mechanisms of Robot Programming by Demonstration*, 54(5):409–413, 2006.

R. Aler, O. Garcia, and J. M. Valls. Correcting and improving imitation models of humans for robosoccer agents. *Evolutionary Computation*, 3(2-5):2402–2409, 2005.

A. Alissandrakis, C. L. Nehaniv, and K. Dautenhahn. Towards robot cultures? *Interaction Studies: Social Behaviour and Communication in Biological and Artificial Systems*, 5(1):3–44, 2004.

A. Alissandrakis, C. L. Nehaniv, K. Dautenhahn, and J. Saunders. Evaluation of robot imitation attempts: comparison of the system's and the human's perspectives. In *Proceedings of the 1st ACM/IEEE International Conference on Human-Robot Interactions (HRI'06)*, 2006.

A. Alissandrakis, C. L. Nehaniv, and K. Dautenhahn. Correspondence mapping induced state and action metrics for robotic imitation. *IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics*, 37(2):299–307, 2007.

R. Amit and M. Mataric. Learning movement sequences from demonstration. In *Proceedings of the 2nd International Conference on Development and Learning (ICDL'02)*, 2002.

B. Argall, B. Browning, and M. Veloso. Learning from demonstration with the critique of a human teacher. In *Proceedings of the 2nd ACM/IEEE International Conference on Human-Robot Interactions (HRI'07)*, 2007.

B. Argall, B. Browning, and M. Veloso. Learning robot motion control with demonstration and advice-operators. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'08)*, 2008.

C. G. Atkeson and S. Schaal. Robot learning from demonstration. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, 1997.

C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11: 11–73, 1997.

J. A. Bagnell and J. G. Schneider. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'01)*, 2001.

D. C. Bentivegna. *Learning from Observation Using Primitives.* PhD thesis, College of Computing, Georgia Institute of Technology, Atlanta, GA, July 2004.

D. C. Bentivegna, A. Ude, C. G. Atkeson, and G. Cheng. Humanoid robot learning and game playing using PC-based vision. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'02)*, 2002.

A. Billard and M. Mataric. Learning human arm movements by imitation: Evaluation of biologically inspired connectionist architecture. *Robotics and Autonomous Systems*, 37(2-3):145–160, 2001.

A. Billard, S. Callinon, R. Dillmann, and S. Schaal. Robot programming by demonstration. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics*, chapter 59. Springer, New York, NY, USA, 2008.

A. G. Billard, S. Calinon, and F. Guenter. Discriminative and adaptive imitation in uni-manual and bi-manual tasks. *Robotics and Autonomous Systems, Special Issue on The Social Mechanisms of Robot Programming by Demonstration*, 54(5):370–384, 2006.

C. Breazeal and B. Scassellati. Robots that imitate humans. *Trends in Cognitive Sciences*, 6(11):481–487, 2002.

C. Breazeal, M. Berlin, A. Brooks, J. Gray, and A. L. Thomaz. Using perspective taking to learn from ambiguous demonstrations. *Robotics and Autonomous Systems, Special Issue on The Social Mechanisms of Robot Programming by Demonstration*, 54(5):385–393, 2006.

B. Browning, L. Xu, and M. Veloso. Skill acquisition and use for a dynamically-balancing soccer robot. In *Proceedings of 19th National Conference on Artificial Intelligence (AAAI'04)*, 2004.

S. Calinon and A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of the 2nd ACM/IEEE International Conference on Human-Robot Interactions (HRI'07)*, 2007a.

S. Calinon and A. G. Billard. What is the teacher's role in robot programming by demonstration? toward benchmarks for improved learning. *Interaction Studies*, 8(24):441–464, 2007b.

A. Chella, H. Dindo, and I. Infantino. A cognitive framework for imitation learning. *Robotics and Autonomous Systems, Special Issue on The Social Mechanisms of Robot Programming by Demonstration*, 54(5):403–408, 2006.

J. Chen and A. Zelinsky. Programing by demonstration: Coping with suboptimal teaching actions. *The International Journal of Robotics Research*, 22(5):299–319, 2003.

S. Chernova and M. Veloso. Confidence-based learning from demonstration using Gaussian Mixture Models. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS'07)*, 2007.

S. Chernova and M. Veloso. Teaching multi-robot coordination using demonstration of communication and state sharing (short paper). In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS'08)*, 2008a.

S. Chernova and M. Veloso. Multi-thresholded approach to demonstration selection for interactive robot learning. In *Proceedings of the 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI'08)*, 2008b.

S. Chernova and M. Veloso. Learning equivalent action choices from demonstration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'08)*, 2008c.

W. S. Cleveland and C. Loader. Smoothing by local regression: Principles and methods. Technical report, AT&T Bell Laboratories, Murray Hill, NJ, 1995.

J. A. Clouse. *On integrating apprentice learning and reinforcement learning.* PhD thesis, University of Massachusetts, Department of Computer Science, 1996.

K. Dautenhahn and C. L. Nehaniv, editors. *Imitation in animals and artifacts.* MIT Press, Cambridge, MA, USA, 2002.

N. Delson and H. West. Robot programming by human demonstration: Adaptation and inconsistency in constrained motion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'96)*, 1996.

Y. Demiris and G. Hayes. Imitation as a dual-route process featuring predictive and learning components: a biologically-plausible computational model. In Dautenhahn and Nehaniv [2002], chapter 13.

Y. Demiris and B. Khadhouri. Hierarchical attentive multiple models for execution and recognition of actions. *Robotics and Autonomous Systems, Special Issue on The Social Mechanisms of Robot Programming by Demonstration*, 54(5):361–369, 2006.

R. Dillmann, M. Kaiser, and A. Ude. Acquisition of elementary robot skills from human demonstration. In *International Symposium on Intelligent Robotic Systems (SIRS'95)*, 1995.

K. R. Dixon and P. K. Khosla. Learning by observation with mobile robots: A computational approach. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'04)*, 2004.

H. Friedrich and R. Dillmann. Robot programming based on a single demonstration and user intentions. In *3rd European Workshop on Learning Robots at ECML'95*, 1995.

A. Garland and N. Lesh. Learning Hierarchical Task Models by Demonstration. Technical Report TR2003-01, Mitsubishi Electric Research Laboratories, 2003.

D. H. Grollman and O. C. Jenkins. Dogged learning for robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'07)*, 2007.

D. H. Grollman and O. C. Jenkins. Sparse incremental learning for interactive robot control policy estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'08)*, 2008.

G. Z. Grudic and P. D. Lawrence. Human-to-robot skill transfer using the spore approximation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'96)*, 1996.

F. Guenter and A. Billard. Using reinforcement learning to adapt an imitation task. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07)*, 2007.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, New York, NY, USA, 2001.

G. Hovland, P. Sikka, and B. McCarragher. Skill acquisition from human demonstration using a hidden markov model. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'96)*, 1996.

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'02)*, 2002a.

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Learning rhythmic movements by demonstration using nonlinear oscillators. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'02)*, 2002b.

T. Inamura, M. Inaba, and H. Inoue. Acquisition of probabilistic behavior decision model based on the interactive teaching method. In *Proceedings of the Ninth International Conference on Advanced Robotics (ICAR'99)*, 1999.

I. Infantino, A. Chella, H. Dzindo, and I. Macaluso. A posture sequence learning system for an anthropomorphic robotic hand. *Robotics and Autonomous Systems*, 42:143–152, 2004.

B. Jansen and T. Belpaeme. A computational model of intention reading in imitation. *Robotics and Autonomous Systems, Special Issue on The Social Mechanisms of Robot Programming by Demonstration*, 54 (5):394–402, 2006.

O. C. Jenkins, M. J. Mataric, and S. Weber. Primitive-based movement classification for humanoid imitation. In *Proceedings of the 1st IEEE-RAS International Conference on Humanoid Robotics*, 2000.

M. Kaiser, H. Friedrich, and R. Dillmann. Obtaining good performance from a bad teacher. In *Programming by Demonstration vs. Learning from Examples Workshop at ML'95*, 1995.

J. Z. Kolter, P. Abbeel, and A. Y. Ng. Hierarchical apprenticeship learning with application to quadruped locomotion. In *Proceedings of Advances in Neural Information Proccessing (NIPS'08)*, 2008.

Y. Kuniyoshi, M. Inaba, and H. Inoue. Learning by watching: Extracting reusable task knowledge from visual observation of human performance. In *IEEE Transactions on Robotics and Automation*, volume 10, pages 799–822, 1994.

S. Lauria, G. Bugmann, T. Kyriacou, and E. Klein. Mobile robot programming using natural language. In *Robotics and Autonomous Systems*, volume 38, pages 171–181, 2002.

J. Lieberman and C. Breazeal. Improvements on action parsing and action interpolation for learning through demonstration. In *4th IEEE/RAS International Conference on Humanoid Robots*, 2004.

A. Lockerd and C. Breazeal. Tutelage and socially guided robot learning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'04)*, 2004.

M. Lopes and J. Santos-Victor. Visual learning by imitation with motor representations. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 35(3):438–449, 2005.

M. J. Mataric. Sensory-motor primitives as a basis for learning by imitation: Linking perception to action and biology to robotics. In Dautenhahn and Nehaniv [2002], chapter 15.

J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems*, 47:79–91, 2004.

C. Nehaniv and K. Dautenhahn. Like me?- measures of correspondence and imitation. *Cybernetics and Systems*, 32(41):11–51, 2001.

C. L. Nehaniv and K. Dautenhahn. The correspondence problem. In Dautenhahn and Nehaniv [2002], chapter 2.

U. Nehmzow, O. Akanyeti, C. Weinrich, T. Kyriacou, and S. Billings. Robot programming by demonstration through system identification. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07)*, 2007.

A. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang. Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*, 2004.

M. Nicolescu and M. J. Mataric. Learning and interacting in human-robot domains. *Special Issue of IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 31(5):419–430, 2001a.

M. N. Nicolescu and M. J. Mataric. Experience-based representation construction: learning from human and robot teachers. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'01)*, 2001b.

M. N. Nicolescu and M. J. Mataric. Methods for robot task learning: Demonstrations, generalization and practice. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'03)*, 2003.

M. Ogino, H. Toichi, Y. Yoshikawa, and M. Asada. Interaction rule learning with a human partner based on an imitation faculty with a simple visuo-motor mapping. *Robotics and Autonomous Systems, Special Issue on The Social Mechanisms of Robot Programming by Demonstration*, 54(5):414–418, 2006.

E. Oliveira and L. Nunes. Learning by exchanging advice. In R. Khosla, N. Ichalkaranje, and L. Jain, editors, *Design of Intelligent Multi-Agent Systems*, chapter 9. Springer, New York, NY, USA, 2004.

M. Ollis, W. H. Huang, and M. Happold. A bayesian approach to imitation learning for robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07)*, 2007.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

N. Pollard and J. K. Hodgins. Generalizing demonstrated manipulation tasks. In *Workshop on the Algorithmic Foundations of Robotics (WAFR'02)*, 2002.

D. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

M. Pomplun and M. Mataric. Evaluation metrics and results of human arm movement imitation. In *Proceedings of the 1st IEEE-RAS International Conference on Humanoid Robotics*, 2000.

P. K. Pook and D. H. Ballard. Recognizing teleoperated manipulations. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'93)*, 1993.

B. Price and C. Boutilier. Accelerating reinforcement learning through implicit imitation. *Journal of Artificial Intelligence Research*, 19:569–629, 2003.

R. P. N. Rao, A. P. Shon, and A. N. Meltzoff. A bayesian model of imitation in infants and robots. In K. Dautenhahn and C. Nehaniv, editors, *Imitation and Social Learning in Robots, Humans, and Animals: Behavioural, Social and Communicative Dimensions*, chapter 11. Cambridge University Press, Cambridge, UK, 2004.

N. Ratliff, J. A. Bagnell, and M. A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*, 2006.

N. Ratliff, D. Bradley, J. A. Bagnell, and J. Chestnutt. Boosting structured prediction for imitation learning. *Proceedings of Advances in Neural Information Processing Systems (NIPS'07)*, 2007.

M. T. Rosenstein and A. G. Barto. Supervised actor-critic reinforcement learning. In J. Si, A. Barto, W. Powell, and D. Wunsch, editors, *Learning and Approximate Dynamic Programming: Scaling Up to the Real World*, chapter 14. John Wiley & Sons, Inc., New York, NY, USA, 2004.

S. Russell. Learning agents for uncertain environments (extended abstract). In *Proceedings of the eleventh annual conference on Computational learning theory (COLT'98)*, 1998.

P. E. Rybski and R. M. Voyles. Interactive task training of a mobile robot through human gesture recognition. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'99)*, 1999.

P. E. Rybski, K. Yoon, J. Stolarz, and M. M. Veloso. Interactive robot task training through dialog and demonstration. In *Proceedings of the 2nd ACM/IEEE International Conference on Human-Robot Interactions (HRI'07)*, 2007.

C. Sammut, S. Hurst, D. Kedzier, and D. Michie. Learning to fly. In *Proceedings of the Ninth International Workshop on Machine Learning*, 1992.

J. Saunders, C. L. Nehaniv, and K. Dautenhahn. Teaching robots by moulding behavior and scaffolding the environment. In *Proceedings of the 1st ACM/IEEE International Conference on Human-Robot Interactions (HRI'06)*, 2006.

S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998.

S. Schaal and D. Sternad. Programmable pattern generators. In *3rd International Conference on Computational Intelligence in Neuroscience*, 1998.

S. Schaal, A. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philosophical Transactions: Biological Sciences*, 358(1431):537–547, 2003.

W. D. Smart. *Making Reinforcement Learning Work on Real Robots.* PhD thesis, Department of Computer Science, Brown University, Providence, RI, 2002.

W. D. Smart and L. P. Kaelbling. Effective reinforcement learning for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'02)*, 2002.

J. Steil, F. Røthling, R. Haschke, and H. Ritter. Situated robot learning for multi-modal instruction and imitation of grasping. *Robotics and Autonomous Systems, Special Issue on Robot Learning by Demonstration*, 2-3(47):129–141, 2004.

A. Steinfeld, T. Fong, D. Kaber, M. Lewis, J. Scholtz, A. Schultz, and M. Goodrich. Common metrics for human-robot interaction. In *Proceedings of the 1st ACM/IEEE International Conference on Human-Robot Interactions (HRI'06)*, 2006.

M. Stolle and C. G. Atkeson. Knowledge transfer using local features. In *Proceedings of IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL'07)*, 2007.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* The MIT Press, Cambridge, MA, London, England, 1998.

J. D. Sweeney and R. A. Grupen. A model of shared grasp affordances from demonstration. In *Proceedings of the IEEE-RAS International Conference on Humanoids Robots*, 2007.

A. L. Thomaz and C. Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Proceedings of 21st National Conference on Artificial Intelligence (AAAI'06)*, 2006.

A. Ude, C. G. Atkeson, and M. Riley. Programming full-body movements for humanoid robots by observation. *Robotics and Autonomous Systems*, 47:93–108, 2004.

M. van Lent and J. E. Laird. Learning procedural knowledge through observation. In *Proceedings of the 1st international conference on Knowledge capture (K-CAP'01)*, 2001.

H. Veeraraghavan and M. Veloso. Teaching sequential tasks with repetition through demonstration (short paper). In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS'08)*, 2008.

S. Vijayakumar and S. Schaal. Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space. In *Proceedings of 17th International Conference on Machine Learning (ICML'00)*, 2000.

R. M. Voyles and P. K. Khosla. A multi-agent system for programming robots by human demonstration. *Integrated Computer-Aided Engineering*, 8(1):59–67, 2001.

M. Yeasin and S. Chaudhuri. Toward automatic robot programming: Learning human skill from visual data. *IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics*, 30, 2000.

B. D. Ziebart, A. Maas, J. D. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of 23rd National Conference on Artificial Intelligence (AAAI'08)*, 2008.