Ph.D. Qualifying Examination

William Curran

April 21, 2014

Contents

1 Introduction

$|S|^{|F|}$ Putting robots into real homes to help those with severe physical disabilities, such as amyotrophic lateral sclerosis (ALS) and quadriplegia, is a long-term goal for our research. Autonomy is one aspect of this goal, however, developing full autonomy for all household tasks for an individual is not possible. We propose to give the user the tools to teach the robot themselves, giving the disabled user both independence and personal customization.

Using a combination of learning from demonstration and reinforcement learning, users can demonstrate to the robot how to perform tasks, and then give feedback via reinforcement learning based on the success of that task. Section 2 and 3 will discuss the current reinforcement learning and learning from demonstration literature, and the tools we can use from each field.

Human-Robot Interaction analyzes how humans can successfully interact with robots in a social environment. Since our user base are people who suffer from severe physical disabilities, they require specially designed interfaces and tools. Section 4 discusses the current Human-Robot Interaction literature, with an emphasis on assistive robots for those with physical disabilities. Section 5 will look at how to combine learning from demonstration with assitive robots. It will analyze how to develop a learning from demonstration algorithm for people who can't provide a good demonstration physically, and who cannot provide timely or quality feedback to guide the learning.

2 Reinforcement Learning

Reinforcement learning is a tool within the field of multiagent or single-agent learning where agents take an action, observe the environment, and receive a reward based on the new environment [96]. Reinforcement learning addresses the problem of finding an optimal policy, $\pi(s, a) = P(a|s)$, that maximizes a reward function, R:

$$J(\pi) = \sum_{s,a} \mu^{\pi}(s)\pi(s, a)R(s, a) \tag{1}$$

where $\mu^{\pi}(s)$ is the probability distribution over policy π (Section 2.3).

To develop this optimal policy, reinforcement learning uses the current state of the agent, s, the action taken, a, the resultant state, s', and a reward corresponding to the system-level success or failure of the state and action, R.

There are three main aspects when defining a stateful reinforcement learner: the policy, the reward function, and the state transitions.

2.1 Policy

The reinforcement learner's policy represents the actions an agent should take in each state. This policy is computed either by a value function representing the quality of a particular action within the state, or a direct search through the policy-space.

2.1.1 Value Function

Learning algorithms based on value functions estimate a value function $V^\pi(s)$ that computes the long-term reward of state s to derive a policy $\pi(s)$ [52]. There are two common value function-based methods to estimate $V^\pi(s)$, policy iteration and temporal difference methods.

Policy iteration and value iteration are model-based approaches that require the transition probabilities $T(s', a, s)$ (the model) and the reward function $R(s, a)$ to calculate the value function. Policy iteration consists of alternating between policy evaluation and policy improvement. Policy evaluation updates the value function by visiting each state in the current policy, and updating the estimate of $V(s)$ [19]. Policy improvement then picks the policy considered optimal. Learning stops when policy improvement converges.

Policy iteration and value iteration guarantee convergence to the optimal solution, assuming that we have a model of the environment and infinite time [88]. Both approaches are nearly exhaustive in the search for an optimal policy, and the time complexity includes the size of the state space and action space. In addition, the model of the environment must be completely accurate in order to avoid accumulation of error. This makes policy iteration difficult to perform in the continuous and large state spaces typically seen in the field of robotics.

Alternatively, temporal difference methods use the reward received at each time step to calculate the difference between the old estimation of the value function and the new estimation. One such example is Q-Learning [96].

In contrast to policy iteration approaches, the temporal difference learner does not perform a search of all successor states. The agent is required to arbitrarily explore previously unseen states or unused actions. In order to find an optimal policy, the agent visits as much of the state-action space as possible. This can result in large and sudden modifications to the current policy. This instability is not desired in robotics, since the robot can be easily damaged. Cautious exploration must be enforced by either avoiding significant changes in the policy (enforcing stability) [82], or strictly disallowing the entering of undesired states [32].

Temporal difference methods also guarantee convergence to the optimal solution, with the same assumptions of infinite time [88]. Since these methods do not require the search of all successor states, the computational complexity is much lower. Temporal difference learning methods are the most widely used methods in reinforcement learning due to the simplicity, minimal amount of computation, and fewer constraints than alternative methods [96].

Value function approaches are compact, simple, and have been proven to find an optimal policy [56]. However, in order to obtain these optimality guarantees, nearly the complete state-action space must be visited, which is often computationally intractable in robotics. In addition, the value function must be discretized when used in a continuous state or action space problem, which can lead to approximation errors. Approximation errors can lead to errors in the value function, which can propagate throughout the policy [31].

2.1.2 Policy Search

Policy search algorithms are the reinforcement learning approach of choice in robotics [31, 56]. Policy search methods use parameterized policies, rather than a value function, essentially performing local explorations through policy space.

Parameterized policies enable reinforcement learning to scale to high dimensional and continuous action spaces by reducing the space of possible policies. Although the localized policy search removes the convenient optimality guarantees of value function reinforcement learning, it includes the benefits of better scalability [31, 56], more intuitive policy initialization with expert knowledge [81], and stability [15].

Given that policy π is parameterized by the set of parameters w, the most general policy representation is:

$$\pi(s) = w^T \theta(s) \tag{2}$$

where $\theta(s)$ is a basis function vector, typically linear in simple problems, and nonlinear with additional free parameters in more complex problems. Alternatively, policies can be represented by a function approximation, such as a neural network [44].

Similar to value function approaches, policy search repeats a three step process: exploration of the state space, evaluation of the policy, and the update of the current policy. The exploration strategy is an important step in policy improvement for all reinforcement learning approaches and was discussed in the previous section.

There are two types of policy evaluation when performing policy search, step-based and episodic. The step-based policy evaluation is identical to the iterative policy evaluation approach used in value function learning. The quality of an action is given by the expected future reward of executing an action in a specific state. Value function approaches can be used in this case, as well as a Monte-Carlo estimation.

The episodic policy evaluation uses the expected return of the entire learning episode to update the parameter vector w. Rather than updating the value function associated with a specific state-action pair, episodic policy evaluation evaluates the policy parameters directly.

Once a policy is evaluated, it needs to be updated based on the resultant evaluation of the state-action pair or the policy parameters. Most policy search research focuses on improving the policy update step with respect to maximizing sample efficiency, minimizing the number of tunable parameters, and increasing learning stability [31]. There are many mathematical approaches for policy update, including policy gradient [14, 57, 81, 93, 102], expectation-maximization [55, 73], and information-theoretic approaches [29, 82].

Policy gradient approaches are direct, in that they directly compute the gradient to maximize policy performance. The REINFORCE algorithm was one of the first policy search algorithms, performs well, and computes policy updates without explicitly computing gradient estimates [102]. However, these approaches require the user to specify a learning rate, which can lead to instability or slow learning if not chosen correctly.

Expectation-Maximization methods remove the need for a learning rate. Variational Inference for Policy Search (VIP) is a typical example. It uses inference to autonomously determine the exploration rate and learning rate [73] but these approaches can potentially have an unstable learning process as well as tends to converge to local minima [31].

Information-theoretic approaches are considered the best for policy search, as they combine the benefits from both policy gradient and expectation-maximization approaches [31]. Information-theoretic approaches bound the distance between the prior policy and the new policy at each update step, enforcing stability in the learning process, and simultaneously updates its policy using expectation-maximization, removing the learning rate [31].

2.2 Reward Function

The reward function encompasses the high-level goal of the system. When an agent takes an action that is good for the system, the reward function ideally returns to the agent a positive reward proportional to how much it helped the system-level reward. Ideally, reinforcement learners also receive a correspondingly smaller reward when their actions are detrimental to the system. In this way agents can iteratively update to converge to the optimal policy according to their reward design.

Reward functions can be given as a binary success or fail value at the end of an episode, and can be given iteratively throughout the simulation. Since success in a robotic task involves many actions, rewarding each action is an ideal iterative way for a reinforcement learner to quickly converge. This approach of guiding the learning process with intermediate rewards is called reward shaping.

2.3 State Transitions

State transitions can be computed using a mathematical model or ignored by learning directly on the robot. Many intractable reinforcement learning problems can be made possible through the use of a model-based approach. Requiring fewer learning examples on the real robot is desirable, as robots need long execution times, manual intervention, and maintenance. However, models must be created from real-world data or computing the physical model of the dynamics. If the model is not accurate, model errors can accumulate and invalidate the learned policy.

3 Learning from Demonstration

Developing a policy using traditional straightforward reinforcement learning does not work well in robotics. Finding an optimal or near-optimal solution requires exploration throughout much of the state space and excessive exploring of the unknown state space risks damaging the robot. This can be avoided by taking small steps during exploration, but this brings the additional problem of taking longer to find an optimal solution. Initializing, or bootstrapping, the policy close to the desired robot behavior removes many of these problems. One particular approach to policy initialization is Learning from Demonstration (LfD).

LfD learns a policy using examples or demonstrations provided by a human or a robotic teacher. These examples are typically state-action pairs that are recorded during the teacher's demonstration. These state-action samples are used to initialize a policy that can then either be directly utilized, or improved using reinforcement learning [7].

There are two main divisions of research within LfD: developing methodology for acquiring the demonstration data, and deriving policies from this data. The data can be acquired from many different sources, such as another robot, another human, a human controlling a robot, or even a human controlling the learner robot. The data acquisition choice heavily affects the difficulty of mapping from real-world data to state-action pairs and will be discussed in Section 3.1.

There are a variety of techniques that LfD algorithms use when deriving a policy. Machine learning directly maps the demonstrated data to a policy to be sampled (Section 3.2.1). This approach attempts to directly learn the teacher's demonstration. On the other hand, reinforcement learning uses the demonstrated data to guide learning, rather than directly learn the demonstration (Section 3.2.2). The demonstrated data can be used to initialize a policy, develop a reward function, and build a state transition function. This approach takes the demonstrated data and attempts to optimize it.

3.1 Data Acquisition

When gathering demonstration data, the choice of demonstrator greatly impacts the complexity. If the teacher is directly controlling the learning robot, the learner's execution can be directly used as demonstration data. However, if the learner is required to observe, the learner must sense the teacher's execution and convert it to demonstration data. This can lead to a large computational overhead. Additionally, if the learner is not physically similar to the teacher, it may also have to convert demonstration data to its own action and state space, adding more computational complexity.

There are four different demonstration approaches at different levels of complexity: teleoperation, mimicking, sensors on the teacher, and simply observing the teacher. Teleoperation removes any complex mapping from the system, making it the most direct approach for data acquisition. However, teleoperation assumes that the teleoperator has the ability to properly control the low-level commands of robots. With higher degree of freedom robots, this teleoperation can become very difficult to perform accurately.

Teleoperation is one of the most commonly used data acquisition techniques [7] and is usually performed with a joystick. Some examples include using teleoperation demonstrations to control an inverted autonomous helicopter [74], robotic grasp modeling [97], object grasping [85], and simulated driving [2].

High-level teleoperation such as speech recognition and kinesthetic teaching can also be employed with great success. Natural language processing has been used to allow a human teacher to verbally teach the robot how to navigate a miniature town [62], to clarify ambiguous demonstrations [20], and general task training [89]. Kinesthetic teaching involves humans physically moving the robot's passive joints through the desired motions [3].

Mimicking requires the learner to observe the demonstrator, convert the demonstrator's execution to actions, and then execute those actions. This assumes the learner's set of actions are the same as the demonstrator's set of actions. Therefore, the only additional computation comes from observing and mapping from the teachers movements to demonstration data. Mimicking has been used analyze how biological animals learn through imitation [33], navigation tasks [72, 75], and to learn arm gestures [78].

By adding sensors directly onto the demonstrator, the learner no longer has to observe the teacher. Rather than having the learner observe the teacher, the teacher records its own execution with sensors. This is similar in computational complexity to mimicking, except the extra computation involves mapping from the demonstration data to the learner, rather than from the demonstrator to the demonstration data. This is especially useful when the application requires precise measurements of the teacher. Researchers have used this technique to teach robots tennis swings [47] and walking patterns [71] by using human teachers wearing sensors.

Lastly, the most general and computationally expensive data acquisition approach involves the robot observing the teacher. The learner must extract the teachers state-action pairs from the observed demonstration. Those state-action pairs then need to be mapped to the learners set of actions. This leads to an imprecise, computationally expensive and less reliable approach than other data acquisition methods [7]. However, this approach is the most general of all techniques. Mimicking requires the learner's set of actions to match the teacher's set of actions, while adding sensors to the teacher requires expensive sensors to accurately obtain demonstration data. Visual features analyzed with computer vision have been used to teach human movement, grasping tasks, and hand gestures [1, 4, 16]. Hybrid approaches combining computer vision with sensors on the teacher or speech recognition has also been successfully used. A force sensing glove on the teacher combined with computer vision motion tracking has been used to teach grasps and other motor skills [63, 66, 100] and visual cues combined with speech recognition has been used to learn human gestures and object tracking [34, 95].

The choice of approach is situationally dependent on the tools available and the users. Teleoperation is useful in low DoF robots and requires only some training and low cost tools. Mimicking, sensors on the teacher, and observation all require another robot or human to first perform the task, which may not always be possible.

3.2 Policy Derivation

Now that the learner has acquired demonstration data from the teacher, it can develop a policy. Many policy derivation approaches exist, but this paper will focus on machine learning and reinforcement learning. Machine learning attempts to develop a policy as close to the teacher's demonstration as possible while generalizing to states unseen by the teacher, and reinforcement learning uses the teacher's policy to initialize the policy, help develop the reward function, or build a state transition function, and then applies reinforcement learning techniques to optimize performance of the task.

3.2.1 Machine Learning

Machine learning approaches use function approximation to develop a mapping from the demonstration states to actions taken. There are two ways to accomplish this: classification and regression. Using both techniques, the learner can reproduce the teacher's policy and generalize to states not encountered by the teacher. These function approximations take the current state as input, and output a action. The key difference between classification and regression is small. Classification takes the set of state inputs and outputs discrete class labels, and regression takes the set of state inputs and outputs a continuous value. In the terms of policy derivation, this maps to discrete or continuous actions. Therefore, classification works well in applications where the robot executes high-level commands, such as "turn the knob" or "open the door", or the action space can be accurately discretized. Regression is a more general technique and works well with low-level motor primitives which need a continuous input. Each approach has benefits and issues, as we will now discuss.

Classification techniques require that the action space be discretized into a specific number of actions. This greatly reduces the generality of the approach. If the action space is not discretized enough, the policy will not adequately match the teacher's demonstration. If the action space is too discretized, the policy may not converge and will need more data. However, if the actions are discretized correctly, this approach can learn a policy with much less data than with regression.

Classification approaches use low-level actions include controlling a simulated car using Gaussian Mixture Models and demonstration requests for unknown parts of the state space [24], obstacle avoidance using interactive teaching methods [49], and using bio-inspired memory maps and k-Nearest Neighbors to construct dynamically sized action-selection mechanisms [91].

High-level actions are useful when sequences of low-level actions have already been developed. It is much easier to create a policy to accomplish a large task when the action selections represent a sequence of already learned low-level actions. This approach has been used to perform a box sorting task by classifying gestures representing prior learned sequences of actions [90], start a collaborative social dialogue with a robot and a human to learn a button pressing task [65], and to classify behaviors to perform a multi-robot ball rolling task by dynamically determining the need for new demonstrations [25].

Regression techniques are more general than classification methods, as the outputs are continuous actions. Different choices arise when using regression. Rather than choosing how to discretize the action space, in regression the designer must choose when to perform the policy function approximation. If the approximation is done during run time, the regression does not need to expend computation to discover state-action pairs in states that are never visited. However, all of the training data must be stored in memory. This is known as Lazy Learning [11]. If the approximation is done before run time, the entire function approximation must be performed using the entire demonstration data set, including states that will never end up being visited in practice. This is much more computationally expensive, but is generally performed off-line.

Lazy Learning techniques have been successfully used in practice. Some examples include using human policy critiquing to successfully intercept a ball [6], learning rhythmic patterns that robustly cope with external perturbations [48], and learning biped walking patterns [71]. In large problems with a continuous action and state spaces, the amount of demonstration data to process at every time step becomes too computationally complex for real-time robotic movement.

There are many approaches that perform policy function approximation before run time, with the most common being Neural Networks [1, 35, 68, 84]. Neural Network function approximation has been used to teach a robot the game of Kendama, a game similar to ball in a cup [68], control ALVINN (Autonomous Land Vehicle In a Neural Network) to drive in single-lane paved and unpaved roads as well as multi-lane lined and unlined roads [84], performing a peg in a hole task by preprocessing bad demonstrations to make them suitable for learning [35] and learning human full-body motion kinematic patterns [1]. Other approaches have used Gaussian Mixture Regression to teach human gestures to a humanoid robot via sensors on the teacher [22] and Sparse On-Line Gaussian Processes to teach an AIBO robot soccer skills through teleoperation [43]. Even though function approximation of the entire dataset is a computationally complex approach, it moves the processing from being real-time on the robot to before the execution, making this approach more ideal for real-time robotics in complex problems.

3.2.2 Reinforcement Learning

As mentioned in Section 2, reinforcement learning is a tool within the field of multiagent or single-agent learning where agents take an action, observe the environment, and receive a reward based on the new environment [96]. This approach requires exploration in order to improve upon the current policy, but in robotics exploration is hazardous to the robot. Alternatively, the robot can learn in simulation, but that requires an accurate state transition model of the world, $T(s'|a, s)$. When applying reinforcement learning with LfD, the teacher's demonstration can help with both approaches. If the robot is learning in the real world, the teacher's demonstration can be used to initialize the reinforcement learning policy, and therefore learning can take small exploratory actions to improve upon this already accurate policy. If the robot is learning in simulation, the teacher's demonstration can be used to both initialize the policy, and be used as data to construct a state transition function. Reinforcement learning approaches include value function and policy search, which have both been explained in Section 2. In addition to policy initialization and state transition derivation, the teacher's demonstration data can be used to assist in engineering the reward function in both standard and inverse reinforcement learning.

In standard reinforcement learning, the reward function is hand-created by the user. Giving high rewards to states around the goal, and low rewards to states around obstacles are intuitive choices, but it leads to a sparse reward function. States that are not directly nearby obstacles or the goal are arbitrarily explored, and can lead to damage to the robot. Teacher demonstrations can be used to prevent blind exploration caused by sparse reward functions by showing the robot rewarded areas of the state space [94]. Learners have also been taught to ask for demonstration help when they are in an arbitrary area of the state space [21, 26] or exchange advice with other agents [77]. Lastly, the user can weigh the reward function based off of whether the teacher had visited the state [77]. All of these approaches attempt to alleviate the issues of arbitrary action selection and sparse reward functions.

An alternative approach is to not build the reward function manually. Inverse Reinforcement Learning is a field within Reinforcement Learning that learns the reward function. LfD can assist in learning the reward function by analyzing the teacher's demonstrations. Approaches include simply associating higher rewards to states similar to those during the demonstration [12], developing a reward function based off of the similarity between the demonstrated and learner executed policy [79], and learning rewards based purely on human feedback [98].

4 Human-Robot Interaction

Human-Robot Interaction (HRI) is a field of study dedicated to understanding, designing, and evaluating robotic systems for use by or with humans [42]. This interaction can be remote or proximate. This paper will focus on the proximate interactions, where robots are physically and directly working with humans. In this collaborative setting, humans generally treat robots as they would other people [86]. Therefore, in order to understand how to socially collaborate with robots, it's important to understand the different forms of interactions.

4.1 Understanding Human-Robot Interactions

One step in understanding the interactions between the human and robot is to understand the level of autonomy of the robot. In a slightly autonomous robot, the human does most of the work through the robot, and the research focus is placed on developing an easy-to-use interface for the human. In a highly autonomous robot, the human and the robot work together to perform a task. However, the robot needs to assist efficiently enough such that the human doesn't simply perform the task.

Historically, autonomy was implemented using one of two approaches. Sense-Plan-Act was an autonomy approach to model decision making. A robot sensed the environment, planned out it's actions, and then acted upon them. This model represents high-level reasoning and assumes a perfect world model with no external perturbations. One early example of using Sense-Plan-Act was applying the STRIPS [39] planning algorithm to Stanford's CART robot [70]. However, planning requires high-level logic and knowledge representation.

On the other end of autonomy approaches is Behavior-Based Robotics, or reactive reasoning. Rather than planning every move, behavior-based robots react directly to sensory inputs. This approach does not need a model of the world, but is also limited in intelligence. A behavior-based robot is only able to react to the environment is can currently sense, meaning that it is unable make high-level or long-term decisions [8].

Current research uses a hybrid of Sense-Plan-Act and Behavior-Based Robotics, known as shared autonomy. In these systems, the high-level planning is separated from the low-level reactions [18]. One famous example is AuRA [9]. In AuRA, high-level goals are given by the user, and then converted to plans and goals by the algorithm. Once these plans and goals are computed, they are sent to the robot for execution, and reactive reasoning takes over. Alternative approaches include a more cooperative human-robot interaction. High-level reasoning can be taken by the human, while low-level reactive behaviors can be executed by the robot. Autonomous robots are efficient at performing difficult low-level manipulation and navigation tasks, but they are limited to whatever knowledge they currently possess, or what knowledge their sensors can give them. Humans are much more efficient at being able to classify and detect objects in the world, as well as performing high-level reasoning. Humans operators can provide navigation, perception and manipulation assistance to an autonomous robot, essentially performing the high-level vision or planning tasks [10, 37, 41, 83]. With humans performing the high-level tasks and the robot performing low-level reactive reasoning, robots become more of a collaborative assistant than a tool.

Another step in shaping the interactions between humans and robots is understanding the way information is exchanged. Typically in HRI, information is exchanged with the robot through visual interfaces, gestures, or speech [42]. Visual interfaces are typically used in a non-proximate teleoperation setting, such as using a robot to assist in search and rescue [13, 51] or optimizing complex navigation tasks [76].

Gestures and speech are more socially involved and require the robot to be able to analyze and understand the semantic meaning. Although this approach is difficult, it has been argued that social interfaces are the most effective ways of communication with robots [86]. The social impact of robots have been analyzed when bringing a TV remote control to a human subject tired from work [30], playing with children diagnosed with autism [87], starting a dialog with a human subject to perform home services [46], and using gesture and vocal contextual cues to process fragmented human commands [80].

4.2 Telepresence Robots

Telepresence robots can allow a user to interact with remote people through video and audio communication. The user can socialize, move, and interact with others through a robotic surrogate. This is especially important for those who cannot leave their home due to temporary sickness, severe physical disorders, or simply out of town for a meeting.

Control systems are generally teleoperated, but as mentioned previously, gestures and speech are more socially important. These natural human interfaces have been a focus of research for people who cannot socially interact with the outside world due to physical disabilities [36].

There has been many applications of telepresence robots, such as health care, schooling, and office environments. In the health care field, it has been shown that 2/3 of people surveyed would rather have their own physician see them via telepresence than see a different physician [64]. In schooling, robots have been used to allow young hospitalized children to still go to school. PEBBLES was developed with a mobile base and arm attachments to give children the ability to participate in a classroom environment [101]. Lastly, it is common for geographic distance to separate teams in a work environment. QB [5], Texai [103] and VGo [27] were developed to allow coworkers to visit and communicate to each other via telepresence. This decreases the amount of traveling employees have to perform and allows immediate access to where they are needed [58]. All of the robots mentioned above are designed for those who are able-bodied and can teleoperate to a fine degree of accuracy.

4.3 Assistive Robotics

Assistive robotics is the most wide spread application of HRI. Robots can be used to help those who are mentally or physically disabled, as well as those with learning disabilities. In this application, robots are typically working directly with a human to provide a needed service. However, humans also need to trust that the robot will not injure them or cause undue damage to their home. This is especially true of humans who are already vulnerable due to a severe disability.

Assistive robots have been used to help those with severe physical challenges, such as those who are visually impaired, the elderly, or individuals requiring physical therapy. The goal of these works are to increase the set of tasks those with disabilities can independently perform. For those who are visually impaired, assistive robots have provided navigational assistance in unknown environments [59, 61]. Assistive robotics have also helped the elderly and those in physical therapy with physical mobility assistance [67, 69, 92, 104]. A robotic wheelchair has been developed for those who can no longer operate a standard powered wheelchair [104]. Additionally, since assitive robots have the ability to execute exact force and movement actions, they are beginning to be used in physical therapy applications to assist in limb rehabilitation [67, 92].

One of the most general uses for assistive robots is manipulation assistance. Many of the physical disability cases noted involve the need for assistance with obtaining or moving objects. Since robots can be fitted with a gripper, they can be used in many applications, such as eating, personal hygiene, pushing buttons, turning knobs, flipping switches, and more. Additionally, robotic arms need not only be attached to a mobile robot [17], but can be mounted in ideal locations, such as on a wheelchair [38, 45, 60] or a workstation [28, 99]. For robotic manipulation assistance, algorithm designers must be risk averse and take care to not injure those requiring help. Rather than using autonomous or semi-autonomous control, many applications are controlled via teleoperation, with one using telemanipulation utilizing both speech and gestures [54].

5 Learning Techniques for Individuals with Physical Disabilities

Using assistive HRI for those with extreme disabilities, such as amyotrophic lateral sclerosis (ALS) and quadriplegia, poses entirely new problems. These individuals have minimal ability to move and speak, and need extended care all day, every day. The strain on families needing to take care of these individuals is enormous, and it has been shown that inexperienced family caregivers use prescription drugs for depression, anxiety, and insomnia two to three times more often than the average population [40]. Robotics can be used to assist those with extreme disabilities and remove much of this burden on the family.

As mentioned previously, much research has been focused on visual displays, gestures and speech. However, these individuals have minimal ability to move and speak, and cannot take advantage of most of these interfaces.

“Robots for Humanity” applied a shared autonomy approach to help Henry Evans, a mute quadriplegic [23]. By developing interfaces and tools Mr. Evans could use, they were able to give him the dignity of performing day to day activities independently. Mr. Evans was able to shave, scratch his face, and even give out candy during Halloween. This vastly increased his quality of life.

Those suffering from ALS and quadriplegia need robots in the world now, and cannot wait for full autonomy of every task. We promote using a shared autonomy approach (Section 4.1). We can separate the low-level reactivity from the higher-level reasoning, and give the higher-level reasoning task to the user. This gives the user far more control over the robot, relieving many of the issues of pure autonomy.

We will combine this approach with learning from demonstration. Using demonstration to initialize reinforcement learning provides supervised training data of what actions to perform in states that are encountered (Section 3). Using this initialization, the robot can perform the task to a small degree, and the user can take the role of a teacher, giving feedback (Section 2). However, our user base is people who suffer from severe physical disabilities. These individuals require specially designed interfaces and tools (Section 4.2, 4.3). This work will look at how to apply learning from demonstration for people who can't provide a good demonstration physically, and who cannot provide timely feedback to guide the learning.

5.1 Learning from Demonstration

Learning from demonstration typically assumes an able-bodied user, as well as a demonstration that is close to optimal. It is unlikely that we will achieve close to optimal demonstrations consistently in this work. However, there are a few currently existing approaches that can assist with this. REPS, an information-theoretic policy search algorithm (Section 2.1.2) algorithmically decides the quality of data points based on entropy, and weighs them accordingly. We can modify these weights to include an analysis of consistency when performing multiple demonstrations. Additionally, we can assume that our user is in proximity to the robot often and can provide the learner feedback that evaluates its performance. This can applied to the reward function using inverse reinforcement learning (Section 3.2.2). Minimizing the amount of feedback required and adding an easy to use interface will be an important aspect of this research.

We will first test the interface for users without disabilities. This test will take advantage of the direct first-person control and the usability of the Oculus Rift (Figure 1a) and Razer Hydra (Figure 1b) control systems. The Oculus Rift allows a user to see through the sensors of the robot, while the Hydra can compute the exact location and orientation of controllers in your hand, allowing the user direct control over robotic arms. Using these tools the user can teach the robot through demonstration, initializing the reinforcement learning algorithm to a fine level of detail.

The interface for users with disabilities will use a less direct approach. The robot will come with basic learned autonomous functionality, such as turning knobs, movement, and picking up objects. First, the user will give the robot a sequence of these high level actions to accomplish the overall task. The robot will simulate itself doing the task. We will give the user a movie reel style interface, allowing rewinding and fast forwarding of the robots simulation. The user can then provide feedback on a action or set of actions at their own speed. This shared autonomy approach allows the users to build custom routines for the robot to perform in a timely manner without direct knowledge of the reinforcement learning algorithm.

We will follow this research with a user study analyzing the effectiveness of each of the approaches.

(a) Oculus Rift

(b) Razer Hydra

(c) Google Glass

5.2 Semantic Mapping with Glass

Google Glass (Figure 1c) is a wearable computer with a head-mounted display and camera. It can be used hands-free through voice commands, making it ideal for users with disabilities. It can be used as a computer vision tool to classify objects in the world, however, without a personal database, it cannot know the semantic meaning of all of the objects a user may want to classify.

This next focus of research will involve developing a shared autonomy approach between the user, Google Glass, and the robot. The user can label objects using the Glass interface, and have that semantic mapping wirelessly sent to the robot. This semantic mapping can then be used as additional information when learning from demonstration and when giving tasks to the robot [50]. Additionally, it can be added to the the teleoperation interface when learning and the task interface when giving commands.

For example, the robot can easily positively classify a desk, but it cannot know whose desk it is, or what the desk is used for. The user can tell Google Glass that this particular desk is his work desk, and a specific book is his class textbook. When the user later gives a task to the robot to bring the textbook to his work desk, the robot will already have this semantic information.

5.3 Multi-Robot Coordination

The Robot Interactive Display Environment (RIDE) was developed by Karulf et al. to satisfy the need of a robotic control interface that allows a single user to effectively control a large number of robots [53]. In RIDE, operators are able to switch between direct control of a robot and supervisory control over all robots. This allows the operators as much control over the robots as the situation warrants.

RIDE was originally created to allow one user to control many robots directly, or take a supervisory role. However, the current supervisory role can be expanded to include giving high-level goals, such as bringing the user a book. Combining the abilities learned from demonstration and the semantic information obtained during object classification, we can build a new interface allowing users to give high-level goals to a group of robots.

When using RIDE to assign tasks, the user will be able to select one or more robots, and then select an object that has been given a semantic mapping. The

RIDE interface will display to the user tasks that had been learned which involve interaction with this object and the selected number of robots. The selected robots will then coordinate to accomplish the task given to them by the user. These goal-based tasks will use previously-learned abilities from demonstrations. For example, we have two robots, a PR2 and Turtlebot. The PR2 is large and slow, but has limbs, while the Turltebot has no ability to interact with the environment, but is fast. Through learning by demonstration the PR2 has learned to pick up a book, and has learned to set it down, and the Turlebot has learned to move from one location to another. If the user gave these robots a task to bring a book to himself, they can coordinate these smaller tasks together to perform the larger goal. The PR2 can put the book on the Turtlebot, and the Turtlebot can deliver it.

To accomplish this, each robot will know what tasks it is capable of performing and how well it performs the task. They will will be able communicate this information with each other and coordinate. The previous example required three tasks: picking up book, setting down book, and moving. The PR2 will know it is efficient at the first two tasks, but slow at the third, whereas the Turtlebot knows it cannot perform the first two tasks, but performs the third task well. Applying this coordination allows simple robots to be able to perform much more complicated tasks when cooperating.

6 Conclusion

Those suffering from ALS and quadriplegia need robots in the world now, and cannot wait for full autonomy of every task. This work intends to help those suffering from severe physical disabilities by giving them the ability to teach the robot themselves, as well as easily give positive and negative feedback. By using our shared autonomy approach we separate the low-level reactivity from the higher-level reasoning, and give the higher-level reasoning task to the user.

Additionally, this work helps reduce the cost to the end-user. By using multi-robot coordination, an expensive PR2 ($500,000), can be replaced with a small turtlebot and a mobile arm ($5,000 to $20,000). This cost reduction will help bring assistive robots into the home for a greater number of people.

This work also helps further the state of the art in reinforcement learning by introducing a new Learning from Demonstration technique that uses human demonstration from non-experts. This requires robots to learn from human demonstrations, even when those demonstrations are highly suboptimal. It also furthers the field in robot coordination, allowing a robot to evaluate its own performance, and use that information to efficiently coordinate.

With these new interfaces and tools, individuals with disabilities will be able to accomplish day-to-day tasks without human assistance. The lack of a human assistant performing the task and the addition of positive experiences like teaching, doing it yourself, and being more independent gets us closer to our goal of using robots to positively help those with physical disabilities.

References

[1] Ale Ude A, Christopher G. Atkeson A, and Marcia Riley D. Programming full-body movements for humanoid robots by observation.

[2] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 1–, New York, NY, USA, 2004. ACM.

[3] B. Akgun, M. Cakmak, J. W. Yoo, and A.L. Thomaz. Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective. In *Proceedings of the International Conference on Human-Robot Interaction (HRI)*, 2012.

[4] R. Amit and M. Matari. Learning movement sequences from demonstration. In *Development and Learning, 2002. Proceedings. The 2nd International Conference on*, pages 203–208, 2002.

[5] Anybots. Qb, 2012.

[6] Brenna Argall, Brett Browning, and Manuela Veloso. Learning by demonstration with critique from a human teacher. In *Proceedings of the ACM/IEEE International Conference on Human-robot Interaction*, HRI '07, pages 57–64, New York, NY, USA, 2007. ACM.

[7] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robot. Auton. Syst.*, 57(5):469–483, May 2009.

[8] Ronald C. Arkin. *An Behavior-based Robotics.* MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[9] Ronald C. Arkin and Tucker Balch. Aura: Principles and practice in review. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:175–189, 1997.

[10] J.A. Atherton and M.A. Goodrich. Perception by proxy: Humans helping robots to see in a manipulation task. In *Human-Robot Interaction (HRI), 2011 6th ACM/IEEE International Conference on*, pages 109–110, March 2011.

[11] Christopher G. Atkeson, Andreww. Moore, and Stefan Schaal. Locally weighted learning. *Artificial Intelligence Review*, pages 11–73, 1997.

[12] Christopher G. Atkeson and Stefan Schaal. Robot learning from demonstration, 1997.

[13] M. Baker, R. Casey, B. Keyes, and H.A. Yanco. Improved interfaces for human-robot interaction in urban search and rescue. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 3, pages 2960–2965 vol.3, Oct 2004.

[14] Jonathan Baxter, Lex Weaver, and Peter Bartlett. Direct gradient-based reinforcement learning: I. gradient estimation algorithms. Technical report, National University, 1999.

[15] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 2nd edition, 2000.

[16] Aude Billard and Maja Mataric. Learning human arm movements by imitation: Evaluation of a biologically-inspired connectionist architecture. *Robotics and Autonomous Systems*, 941:1–16, 2001.

[17] G. Bolmsjo, H. Neveryd, and H. Eftring. Robotics in rehabilitation. *Rehabilitation Engineering, IEEE Transactions on*, 3(1):77–83, Mar 1995.

[18] R. Peter Bonasso, David Kortenkamp, David P. Miller, and Marc Slack. Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:237–256, 1995.

[19] Darius Braziunas. Pomdp solution methods. Technical report, 2003.

[20] Cynthia Breazeal, Matt Berlin, Andrew G. Brooks, Jesse Gray, and Andrea Lockerd Thomaz. Using perspective taking to learn from ambiguous demonstrations. *Robotics and Autonomous Systems*, 54(5):385–393, 2006.

[21] M. Cakmak and A. L. Thomaz. Designing robot learners that ask good questions. In *Proceedings of the International Conference on Human-Robot Interaction (HRI)*, 2012.

[22] S. Calinon and A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, pages 255–262, March 2007.

[23] Tiffany L. Chen, Matei T. Ciocarlie, Steve B. Cousins, Phillip M. Grice, Kelsey P. Hawkins, Kaijen Hsiao, Charles C. Kemp, Chih-Hung King, Daniel A. Lazewatsky, Adam Leeper, Hai Nguyen, Andreas Paepcke, Caroline Pantofaru, William D. Smart, and Leila Takayama. Robots for humanity: Using assistive robotics to empower people with disabilities. *IEEE Robot. Automat. Mag.*, 20(1):30–39, 2013.

[24] Sonia Chernova and Manuela Veloso. Confidence-based policy learning from demonstration using gaussian mixture models. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '07, pages 233:1–233:8, New York, NY, USA, 2007. ACM.

[25] Sonia Chernova and Manuela Veloso. Teaching multi-robot coordination using demonstration of communication and state sharing. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '08, pages 1183–1186, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[26] J. Clouse. On integrating apprentice learning and reinforcement learning. Technical report, Amherst, MA, USA, 1997.

[27] VGo Communications. Vgo, 2012.

[28] J.L. Dallaway, R.D. Jackson, and P.H.A. Timmers. Rehabilitation robotics in europe. *Rehabilitation Engineering, IEEE Transactions on*, 3(1):35–45, Mar 1995.

[29] C. Daniel, G. Neumann, and J. Peters. Hierarchical relative entropy policy search. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS 2012)*, 2012.

[30] K. Dautenhahn, M. Walters, S. Woods, K. L. Koay, and C. L. Nehaniv. How may i serve you?: A robot companion approaching a seated person in a helping context. In *HRI 06: Proceeding of the 1st ACM SIGCHI/SIGART conference on Humanrobot interaction*, pages 172–179. ACM Press, 2006.

[31] Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013.

[32] Marc Peter Deisenroth, Carl Edward Rasmussen, and Dieter Fox. Learning to control a low-cost manipulator using data-efficient reinforcement learning. In Hugh F. Durrant-Whyte, Nicholas Roy, and Pieter Abbeel, editors, *Robotics: Science and Systems*, 2011.

[33] John Demiris and Gillian M. Hayes. Imitation in animals and artifacts. chapter Imitation As a Dual-route Process Featuring Predictive and Learning Components: A Biologically Plausible Computational Model, pages 327–361. MIT Press, Cambridge, MA, USA, 2002.

[34] Yiannis Demiris and Bassam Khadhouri. Hierarchical attentive multiple models for execution and recognition of actions. In *Robotics and Autonomous Systems*, pages 361–369, 2005.

[35] R. Dillmann, M. Kaiser, and A. Ude. Acquisition of elementary robot skills from human demonstration. In *International Symposium on Intelligent Robotics Systems*, pages 185–192, 1995.

[36] H.M. Do, C.J. Mouser, Ye Gu, Weihua Sheng, S. Honarvar, and Tingting Chen. An open platform telepresence robot with natural human interface. In *Cyber Technology in Automation, Control and Intelligent Systems (CYBER), 2013 IEEE 3rd Annual International Conference on*, pages 81–86, May 2013.

[37] A. Edsinger and C.C. Kemp. Human-robot interaction for cooperative manipulation: Handing objects to one another. In *Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE International Symposium on*, pages 1167–1172, Aug 2007.

[38] K. Edwards, R. Alqasemi, and R. Dubey. Wheelchair-mounted robotic arms: Design and development. In *Biomedical Robotics and Biomechatronics, 2006. BioRob 2006. The First IEEE/RAS-EMBS International Conference on*, pages 613–618, Feb 2006.

[39] Richard Fikes. Strips, a retrospective. *Artif. Intell.*, 59(1-2):227–232, 1993.

[40] Dolores Gallagher, Jonathon Rose, Patricia Rivera, Steven Lovett, and Larry W. Thompson. Prevalence of depression in family caregivers. *The Gerontologist*, 29(4):449–456, 1989.

[41] Ian J. Goodfellow, Nate Koenig, Marius Muja, Caroline Pantofaru, Alexander Sorokin, and Leila Takayama. Help me help you: Interfaces for personal robots. In *Proceedings of the 5th ACM/IEEE International Conference on Human-robot Interaction*, HRI '10, pages 187–188, Piscataway, NJ, USA, 2010. IEEE Press.

[42] Michael A. Goodrich and Alan C. Schultz. Human-robot interaction: A survey. *Found. Trends Hum.-Comput. Interact.*, 1(3):203–275, January 2007.

[43] D.H. Grollman and O.C. Jenkins. Sparse incremental learning for interactive robot control policy estimation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3315–3320, May 2008.

[44] Verena Heidrich-Meisner and Christian Igel. Neuroevolution strategies for episodic reinforcement learning. *J. Algorithms*, 64(4):152–168, October 2009.

[45] Michael Hillman, Karen Hagan, Sean Hagan, Jill Jepson, and Roger Orpwood. The weston wheelchair mounted assistive robot - the design story. *Robotica*, 20(2):125–132, March 2002.

[46] Jin-Hyuk Hong, Youn-Suk Song, and Sung-Bae Cho. A hierarchical bayesian network for mixed-initiative human-robot interaction. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3808–3813, April 2005.

[47] A.J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 2, pages 1398–1403, 2002.

[48] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning rhythmic movements by demonstration using nonlinear oscillators. In *Proceedings of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS2002*, pages 958–963, 2002.

[49] 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.

[50] Z.S.S. Jini. Semantic image retrieval using relevance feedback and reinforcement learning algorithm. In *I/V Communications and Mobile Network (ISVC), 2010 5th International Symposium on*, pages 1–4, Sept 2010.

[51] M. Waleed Kadous, Raymond Ka-Man Sheh, and Claude Sammut. Effective user interface design for rescue robotics. In *Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-robot Interaction*, HRI '06, pages 250–257, New York, NY, USA, 2006. ACM.

[52] Shivaram Kalyanakrishnan and Peter Stone. An empirical analysis of value function-based and policy search reinforcement learning. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '09, pages 749–756, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[53] Erik Karulf, Marshall Strother, Parker Dunton, and William D. Smart. RIDE: Mixed-mode control for mobile robot teams. In *Proceedings of the 6th ACM/IEEE International Conference on Human-Robot Interaction (Late Breaking Papers Track)*, Lausanne, Switzerland, 2011.

[54] Zunaid Kazi, Shoupu Chen, Matthew Beitler, Daniel Chester, and Richard Foulds. Speech and gesture mediated intelligent teleoperation. In *Assistive Technology and Artificial Intelligence*, volume 1458 of *Lecture Notes in Computer Science*, pages 194–210. Springer Berlin Heidelberg, 1998.

[55] Jens Kober and Jan Peters. Policy search for motor primitives in robotics. *Mach. Learn.*, 84(1-2):171–203, July 2011.

[56] Jens Kober and Jan Peters. Reinforcement learning in robotics: A survey. In *Reinforcement Learning*, volume 12 of *Adaptation, Learning, and Optimization*, pages 579–610. Springer Berlin Heidelberg, 2012.

[57] Nate Kohl and Peter Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2619–2624, May 2004.

[58] Annica Kristoffersson, Silvia Coradeschi, and Amy Loutfi. A review of mobile robotic telepresence. *Adv. in Hum.-Comp. Int.*, 2013:3:3–3:3, January 2013.

[59] Vladimir Kulyukin, Chaitanya Gharpure, John Nicholson, and Grayson Osborne. Robot-assisted wayfinding for the visually impaired in structured indoor environments. *Auton. Robots*, 21(1):29–41, August 2006.

[60] H. Hok Kwee. Integrated control of manus manipulator and wheelchair enhanced by environmental docking. *Robotica*, 16(5):491–498, September 1998.

[61] Gerard Lacey and Kenneth M. Dawson-Howe. The application of robotics to a mobility aid for the elderly blind. *Robotics and Autonomous Systems*, 23(4):245–252, 1998.

[62] Stanislao Lauria, Guido Bugmann, Theocharis Kyriacou, and Ewan Klein. Mobile robot programming using natural language. *Robotics and Autonomous Systems*, 38(3-4):171–181, 2002.

[63] J. Lieberman and C. Breazeal. Improvements on action parsing and action interpolation for learning through demonstration. In *Humanoid Robots, 2004 4th IEEE/RAS International Conference on*, volume 1, pages 342–365 Vol. 1, Nov 2004.

[64] Ellison LM, Nguyen M, Fabrizio MD, Soh A, Permpongkosol S, and Kavoussi LR. Postoperative robotic telerounding: A multicenter randomized assessment of patient outcomes and satisfaction. *Archives of Surgery*, 142(12):1177–1181, 2007.

[65] A. Lockerd and C. Breazeal. Tutelage and socially guided robot learning. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3475–3480 vol.4, Sept 2004.

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

[67] P.S. Lum, C.G. Burgar, P.C. Shor, M. Majmundar, and M. Van der Loos. Robot-assisted movement training compared with conventional therapy techniques for the rehabilitation of upper-limb motor function after stroke. In *Archives of Physical Medicine and Rehabilitation*, 2002.

[68] Hiroyuki Miyamoto, Stefan Schaal, Francesca Gandolfo, Hiroaki Gomi, Yasuharu Koike, Rieko Osu, Eri Nakano, Yasuhiro Wada, and Mitsuo Kawato. A kendama learning robot based on bi-directional theory. *Neural Networks*, 9(8):1281–1302, 1996.

[69] Michael Montemerlo, Joelle Pineau, Nicholas Roy, Sebastian Thrun, and Vandi Verma. Experiences with a mobile robotic guide for the elderly. In *Eighteenth National Conference on Artificial Intelligence*, pages 587–592, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[70] H.P. Moravec. The stanford cart and the cmu rover. *Proceedings of the IEEE*, 71(7):872–884, July 1983.

[71] Jun Nakanishi, Jun Morimoto, Gen Endo, Gordon Cheng, Stefan Schaal, and Mitsuo Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems*, 47:79–91, 2004.

[72] U. Nehmzow, O. Akanyeti, C. Weinrich, T. Kyriacou, and S. A. Billings. Robot programming by demonstration through system identification. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 801–806, Oct 2007.

[73] Gerhard Neumann. Variational inference for policy search in changing situations. In *ICML*, pages 817–824. Omnipress, 2011.

[74] Andrew Y. Ng, H. Jin Kim, Michael I. Jordan, and Shankar Sastry. Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*. MIT Press, 2004.

[75] M.N. Nicolescu and M.J. Mataric. Experience-based representation construction: learning from human and robot teachers. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 2, pages 740–745 vol.2, 2001.

[76] Curtis W. Nielsen and Michael A. Goodrich. Comparing the usefulness of video and map information in navigation tasks. In *Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-robot Interaction*, HRI ’06, pages 95–101, New York, NY, USA, 2006. ACM.

[77] Lus Nunes and Eugenio Oliveira. On learning by exchanging advice. *CoRR*, cs.LG/0203010, 2002.

[78] M. Ogino, H. Toichi, M. Asada, and Y. Yoshikawa. Imitation faculty based on a simple visuo-motor mapping towards interaction rule learning with a human partner. In *Development and Learning, 2005. Proceedings. The 4th International Conference on*, pages 148–148, July 2005.

[79] M. Ollis, W.H. Huang, and M. Happold. A bayesian approach to imitation learning for robot navigation. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 709–714, Oct 2007.

[80] D. Perzanowski, A.C. Schultz, W. Adams, and E. Marsh. Goal tracking in a natural language interface: towards achieving adjustable autonomy. In *Computational Intelligence in Robotics and Automation, 1999. CIRA '99. Proceedings. 1999 IEEE International Symposium on*, pages 208–213, 1999.

[81] J. Peters and S. Schaal. Policy gradient methods for robotics. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2219–2225, Oct 2006.

[82] Jan Peters, Katharina Mülling, and Yasemin Altün. Relative entropy policy search. In Maria Fox and David Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2010)*, pages 1607–1612. AAAI Press, 2010.

[83] B. Pitzer, M. Styer, Christian Bersch, C. DuHadway, and J. Becker. Towards perceptual shared autonomy for robotic mobile manipulation. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 6245–6251, May 2011.

[84] Dean A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Comput.*, 3(1):88–97, March 1991.

[85] Polly K. Pook and Dana H. Ballard. Recognizing teleoperated manipulations. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 578–585, 1993.

[86] Byron Reeves and Clifford Nass. *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places.* Cambridge University Press, New York, NY, USA, 1996.

[87] B. Robins, P. Dickerson, and S. Stribling. Robot-mediated joint attention in children with autism: A case study in robot-human interaction. In *Integration Studies.*

[88] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

[89] Paul Rybski, Kevin Yoon, Jeremy Stolarz, and Manuela Veloso. Interactive robot task training through dialog and demonstration. In *2nd ACM/IEEE International Conference on Human-Robot Interaction*, March 2007.

[90] P.E. Rybski and R.M. Voyles. Interactive task training of a mobile robot through human gesture recognition. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 1, pages 664–669 vol.1, 1999.

[91] Joe Saunders, Chrystopher L. Nehaniv, and Kerstin Dautenhahn. Teaching robots by moulding behavior and scaffolding the environment. In *Human-Robot Interaction*, pages 118–125. ACM Press, 2006.

[92] N. Shirzad and H.F.M. Van der Loos. Error amplification to promote motor learning and motivation in therapy robotics. In *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pages 3907–3910, Aug 2012.

[93] C. Shu, H. Ding, and N. Zhao. Numerical comparison of least square-based finite-difference (lsfd) and radial basis function-based finite-difference (rbffd) methods. *Comput. Math. Appl.*, 51(8):1297–1310, April 2006.

[94] W.D. Smart and L.P. Kaelbling. Effective reinforcement learning for mobile robots. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 4, pages 3404–3410 vol.4, 2002.

[95] Jochen J Steil, Frank Roethling, Robert Haschke, and Helge Ritter. Situated robot learning for multi-modal instruction and imitation of grasping. *Robotics and Autonomous Systems*, Special Is(47):129–141, 2004.

[96] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning I: Introduction.* 1998.

[97] John D. Sweeney and Rod Grupen. A model of shared grasp affordances from demonstration. In *Proceedings of the IEEE-RAS International Conference on Humanoids Robots, Humanoids07*, 2007.

[98] Andrea L. Thomaz and Cynthia Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, pages 1000–1005. AAAI Press, 2006.

[99] M. Topping. The development of handy 1 a robotic aid to independence for the severely disabled. In *Mechatronic Aids for the Disabled, IEE Colloquium on*, pages 2/1–2/6, May 1995.

[100] Richard M. Voyles. A multi-agent system for programming robots by human demonstration. *Integrated Computer-Aided Engineering*, 8:59–67, 2001.

[101] Laurel A. Williams, Deborah I. Fels, and Jutta Treviranus. Using pebbles to facilitate remote communication and learning.

[102] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Machine Learning*, pages 229–256, 1992.

[103] Inc. Willow Garage. Texai remote presence system, 2010.

[104] HollyA. Yanco. Wheelesley: A robotic wheelchair system: Indoor navigation and user interface. In VibhuO. Mittal, HollyA. Yanco, John Aronis, and Richard Simpson, editors, *Assistive Technology and Artificial Intelligence*, volume 1458 of *Lecture Notes in Computer Science*, pages 256–268. Springer Berlin Heidelberg, 1998.