Automatic Weight Learning for Multiple Data Sources when Learning from Demonstration

Brenna D. Argall, Brett Browning and Manuela Veloso

Abstract— **Traditional approaches to programming robots are generally inaccessible to non-robotics-experts. A promising exception is the *Learning from Demonstration* paradigm. Here a *policy* mapping world observations to action selection is learned, by generalizing from task demonstrations by a teacher. Most Learning from Demonstration work to date considers data from a single teacher. In this paper, we consider the incorporation of demonstrations from multiple teachers. In particular, we contribute an algorithm that handles multiple data sources, and additionally reasons about *reliability* differences between them. For example, multiple teachers could be inequally proficient at performing the demonstrated task. We introduce *Demonstration Weight Learning (DWL)* as a Learning from Demonstration algorithm that explicitly represents multiple data sources and learns to select between them, based on their observed reliability and according to an adaptive *expert learning* inspired approach. We present a first implementation of DWL within a simulated robot domain. Data sources are shown to differ in reliability, and weighting is found impact task execution success. Furthermore, DWL is shown to produce appropriate data source weights that improve policy performance.**

I. INTRODUCTION

As robots become more prevalent within general society, the need for programming techniques that are accessible to non-experts increases. To develop a control *policy*, or mapping from world observation to action selection, traditional approaches first model world dynamics and then derive the policy mathematically. Though theoretically well-founded, these approaches depend heavily on the accuracy of the world model, which requires considerable expertise to develop. Other model-free approaches are in general similarly restricted in use to robotics-experts.

One potential exception is policy development through *Learning from Demonstration (LfD)*, e.g. [7], [12]. Under this paradigm, a teacher first demonstrates a desired behavior to the robot. The robot then generalizes from these examples to derive a policy. Demonstration has the attractive feature of being an intuitive communication medium for humans, who already use demonstration to teach other humans. Since it does not require robotics expertise, demonstration also opens policy development to non-robotics-experts.

B. Argall and B. Browning are with the Robotics Institute, and M. Velsoso the Computer Science Department, at Carnegie Mellon University, Pittsburgh, PA 15213, USA. <bargall,brettb,mveloso>@cs.cmu.edu

This research was sponsored in part by the Boeing Company under Grant No. CMU-BA-GTA-1. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied of the Boeing Company. The authors Brenna Argall and Brett Browning were partially supported by Carnegie-Mellon University in Qatar and the Qatar Foundation.

In this work, we explicitly consider LfD policy development that incorporates demonstration data from multiple sources. We argue that this consideration is particularly relevant for LfD robotics applications in general society, where multiple users of a robot system is likely. Multiple users easily translates into multiple policy developers, and therefore into multiple LfD teachers.

We further posit that all LfD data sources may not be equally reliable. Multiple teachers may differ in their respective abilities to perform a given task. Another possibility is that sources produce data in fundamentally different ways.

We introduce *Demonstration Weight Learning (DWL)* as an algorithm that incorporates data from multiple demonstration sources. DWL considers reliability, by assigning a weight to each data source. The weight is automatically determined and updated by the algorithm, based on learner performance using the LfD policy. Though the algorithm is general to any sort of LfD domain, we focus the scope of this work to motion control tasks.

This paper contributes a first implementation of DWL, in addition to the algorithm introduction. DWL is a policy learning algorithm that explicitly addresses differences in LfD data source reliability. Results from within a simulated robotic domain are presented, and data source weighting is found impact task execution success. Furthermore, DWL is shown to produce appropriate data source weights that improve policy performance.

The remainder of the paper is organized as follows. Section II motivates weighting multiple LfD data sources, while providing related work. The DWL algorithm is presented in Section III, followed by implementation and experimental details in Section IV. Empirical results are presented and discussed within Section V, and in Section VI we conclude.

II. MOTIVATION AND RELATED WORK

We begin with a discussion of Learning from Demonstration and the related work that motivates our consideration of multiple weighted demonstration sources.

A. Learning from Demonstration

During an LfD teacher execution, world state observations and action selections are recorded. Formally, our world consists of states S and actions A, with the mapping between states by way of actions being defined by the 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 teacher demonstration $d_j \in D$ is represented as l_j

pairs of observations and actions such that $d_j = \{(\mathbf{z}_j^i, \mathbf{a}_j^i)\} \in D, \mathbf{z}_j^i \in Z, \mathbf{a}_j^i \in A, i = 0 \cdots l_j$. As no distinction is made within D between the individual executions, for succinctness we notate $(\mathbf{z}_k, \mathbf{a}_k) \equiv (\mathbf{z}_j^i, \mathbf{a}_j^i)$. Within the LfD paradigm, the set D of these demonstrations are provided to the learner. A policy $\pi : Z \rightarrow A$, that selects actions based on observations of the world state, is then derived.

LfD has found success on a variety of robot platforms and applications, e.g. [1], [5]. Key design decisions include the demonstration approach that produces the data, and then how a policy is derived from that data.

Approaches for executing and recording teacher demonstrations range from teleoperating a passive robot platform [14] to recording sensors worn by a human [9]. Our simulated robot is teleoperated while recording from its own sensors, as this minimizes correspondence issues between demonstrator and learner on real platforms. We restrict our scope to human demonstrators, though the algorithm itself is general to any teacher.

The most popular approaches for deriving a policy from demonstration data are to (i) directly approximate the underlying function mapping observations to actions [7], (ii) use the data to determine the world dynamics model $T(s'|s,a)$ [13] or (iii) provide a planner with a learned model of action pre- and post-conditions [12]. Our work derives a policy using the first mapping function approximation approach.

B. Multiple Weighted Data Sources

This work introduces an algorithm that reasons explicitly about demonstration data from multiple sources. Previous work within LfD considers multiple demonstration teachers for the purposes of isolating salient characteristics of the task execution [14]. Demonstration information solicited from multiple other agents speeds up learning [13], and additional data sources address circumstances under which teacher execution is difficult or inefficient [3].

This work additionally considers the unequal reliability of different data sources. We further consider that the reliability of a data source is not static. For example, data based on learner executions may become more reliable as the learner becomes more proficient at reproducing the target behavior. Previous LfD work considers the issue of data worth, for example by actively removing unnecessary or inefficient elements from a teacher demonstration [11], as well as dataset reliability, for example using a confidence measure to identify undemonstrated or ambiguously demonstrated areas of the state space [10].

Our work is unique in combining the two LfD considerations of multiple data sources and reliability. Our DWL algorithm addresses the issues of data source reliability and stochasticity by assigning a weight to each data source and dynamically updating that weight as the policy develops. Data source weighting is accomplished via an expert learning approach, where each data source is treated as an expert. In particular, we adopt a modified version of Exp3 [6], an expert learning algorithm with successful prior robot applications [2], [8]. Unlike typical expert learning approaches, Exp3 does not depend on observing rewards for *all* experts at each decision point. Exp3 thus is able to handle domains where the consequence of only *one* prediction is observed, e.g. domains like robot motion control where predictions execute physical actions.

III. ALGORITHM

In this section we present the Demonstration Weight Learning algorithm. DWL is an LfD approach characterized by the explicit incorporation and weighting of multiple demonstration data sources. Details of the algorithm execution, followed by the dynamic weight update, are presented.

A. Execution Overview

Given a set of demonstration sources C, DWL automatically determines a set of weights $\mathbf{w}$ over the sources. These weights are considered when deriving a policy π from the demonstration data $\mathbf{D}$, and are dynamically updated in response to learner execution performance. Psuedo-code for DWL is presented in Algorithm 1. For succinctness, we define $\psi_i^t \equiv I(c^t =^? c_i)$, as an indicator on whether source c^t is equal to the source c_i.

Algorithm 1 Demonstration Weight Learning (DWL)

```
1: *init* $\mathbf{w} \leftarrow 1$
2: *init* $\pi \leftarrow$ policyDerivation$(\mathbf{D}, \mathbf{w})$
3: **for** $\mathbf{z}^{goal} \in Z$ **do**
4:     **repeat**
5:         *select* $\mathbf{a}^t \leftarrow \pi(\mathbf{z}^t)$
6:         *execute* $\mathbf{a}^t$
7:         *record* $tr \leftarrow \{ \mathbf{z}^t, \mathbf{a}^t, c^t \}$
8:     **until** $\mathbf{z}^t = \mathbf{z}^{goal}$
9:     *adapt* $\mathbf{w}$
10:    *rederive* $\pi \leftarrow$ policyDerivation$(\mathbf{D}, \mathbf{w})$
11: **end for**
12: **return** $\pi$
```

The first phase of DWL consists of teacher demonstration. Demonstrations from multiple sources produce dataset $\mathbf{D}$, where source c_i produces data subset $D_{\psi_i} \in \mathbf{D}$. From the dataset $\mathbf{D}$ an initial policy π is derived (line 2). All data source weights are initialized to 1.

The second phase of DWL consists of learner practice. For each practice run, the learner executes the task until reaching goal state $z^{goal} \in Z$, producing execution trace tr. At each timestep the learner selects action $\mathbf{a}^t$, recommended by a source $c_i \in C$, according to policy $\pi(\mathbf{z}^t)$ (line 5); the details of this selection are provided in Section III-B. This action is recorded in the execution trace tr, along with observation $\mathbf{z}^t$ and data source $c^t = c_i$ (line 7).

Following a learner execution, the data source weights and policy are updated. First, the data source weights $\mathbf{w}$ are adapted based on the learner execution performance (line 9); the details of this adaptation are provided in Section III-C. Then a new policy π is derived from the updated set D and adapted weights $\mathbf{w}$ (line 10).

B. Policy Prediction and Data Source Selection

At each time step, the policy produces a single action for execution. To do so, a data source is sampled at random from a uniform distribution, weighted by source weights $\mathbf{w}$. The action prediction of the selected data source is then the policy output.

To make a prediction, data sources may employ any sort of policy derivation technique appropriate to the underlying data, for example classification or regression techniques. Our specific motion control implementation considers continuous-valued predictions and employs a form of Locally Weighted Learning [4] for regression.

Formally, given observation $\mathbf{z}^t$, action $\mathbf{a}^t$ from source c_i is predicted through an averaging of the data points $(\mathbf{z}_j, \mathbf{a}_j) \in D_{\psi_i^t}$, weighted by their kernelized distance to $\mathbf{z}^t$. Thus,

$$\mathbf{a}^t = \sum_{(\mathbf{z}_j,\mathbf{a}_j)\in D_{\psi_i^t}} \phi\left(\mathbf{z}^t, \mathbf{z}_j\right) \cdot \mathbf{a}_j \ , \tag{1}$$

$$\phi\left(\mathbf{z}^t, \mathbf{z}_j\right) = e^{\left(\mathbf{z}_j - \mathbf{z}^t\right)\Sigma^{-1}\left(\mathbf{z}_j - \mathbf{z}^t\right)^T}$$

where the kernel weights $\phi\left(\mathbf{z}^t, :\right)$ are normalized over j, and Σ^{-1} is a constant parameter scaling each observation dimension to within $[0, 1]$. In our implementation the distance computation is Euclidean, and the kernel is Gaussian.

C. Expert Learning for Selection

Data source selection in DWL takes an *expert learning* approach. Originally proposed by Robbins [15] to solve the k-armed bandits problem, expert learning addresses the issue of choosing between multiple action recommenders, or *experts*. Under this paradigm, executing an action receives reward, for which the recommending expert is credited. An expert's selection probability is determined by its accumulated reward, such that high reward, and therefore good performance, increases the probability of being selected. DWL treats data sources as experts.

Formally, at each decision cycle, t, each of n experts makes a recommendation. The algorithm selects a single expert and executes the corresponding action, resulting in payoff $r^t \in \Re$. After d decision cycles, a sequence of $r^1, r^2, \cdots, r^d$ payoffs have been awarded. The aim of expert learning is to select the best expert over all decision cycles. The learning objective is formulated in terms of *regret*, or the difference between reward r^t of the selected action and reward r_b^t of the action recommended by the best expert. Summed over all decision cycles,

$$Regret = \sum_{t=1}^{d} r_b^t - \sum_{t=1}^{d} r^t. \tag{2}$$

The goal of expert learning is to minimize this total regret.

When actions take a physical form as in robot applications, however, only the reward for the executed action is observed. The rewards that would have been received by the other experts are *not* observable. Learning thus becomes a partial information game. Algorithm Exp3 [6] handles partial information games by scaling reward inversely with selection probability. The effective reward $\hat{r}^t$ earned by selected expert i from reward r^t is thus

$$\hat{r}^t = \frac{r^t}{P(\psi_i^t)} \ , \qquad P(\psi_i^t) = \frac{w_i^t}{\sum_{j=1}^{n} w_j^t} \tag{3}$$

where $P(\psi_i^t)$ is the probability of selecting expert i at decision cycle t and w_i^t is the selection weight of expert i. This compensates for the fact that experts with low probability are infrequently chosen, and therefore have fewer observed rewards. The selection weight of expert i then updates to

$$w_i^t = e^{\hat{r}^t} w_i^{t-1} \tag{4}$$

with weights being initially equal across experts. Note that the exponent product is equivalent to adding $\hat{r}_i^t$ to $\sum_{t=1..d} \hat{r}_i^{t-1}$, and thus represents the cumulative reward received by expert i up to trial t.

D. The Dynamic Weight Update in DWL

To update data source selection weights, DWL models its approach on the Exp3 expert learning algorithm [6]. Similar to Exp3, reward r_i^t is scaled inversely with data source selection probability $P(\psi_i^t)$ (Eq. 3). The DWL algorithm further makes two novel contributions to this weight update, relating to the distribution of expert rewards and determination of the expert selection probabilities.

1) Assigning Individual Reward: The DWL algorithm receives a single reward r in response to a learner execution. This single execution, however, consists of a *sequence* of expert selections, which occur at each timestep. Reward therefore is not received at every decision cycle. Furthermore, multiple experts may contribute to a single execution, and accordingly thus also to the received reward.

To address this issue, reward is *distributed* amongst all experts that contributed to an execution. The contribution of expert i to the execution trace tr is computed as the fractional number of execution points for which i was the recommending expert. This contribution then combines with observed reward r^t, so that the individual reward r_i^t received by expert i is computed according to

$$r_i = r\left(\frac{k_{tr_i}}{k_{tr}}\right) \ , \tag{5}$$

$$k_{tr_i} = \sum_t \psi_i^t, \quad t = 1 \cdots k_{tr}$$

where k_{tr} is the number of execution timesteps in tr and k_{tr_i} is the number of those that selected data source i. This reward is then further scaled by the inverse selection probability of expert i, as in Equation 3.

2) Determining Selection Probabilities: Under DWL, an expert's selection probability is governed by two factors. The first is selection weight, as defined in Section III-C. The second is data *distribution* in the state space. Sources are not assumed to contribute data uniformly across the state-space and, task proficiency aside, should not be trusted to make predictions in areas where they have no support.

To address this, selection probability $P(\psi_i)$ is formulated based on the distance between dataset D_{ψ_i} and the query point, in addition to the selection weight w_i. Concretely, given observation $\mathbf{z}^t$, the probability of selecting source i at timestep t is given by

$$P\left(\psi_i^t\right) = \frac{p\left(\psi_i^t|\mathbf{z}^t\right)}{\sum_{j=1}^{n} p\left(\psi_j^t|\mathbf{z}^t\right)} , \tag{6}$$

$$p\left(\psi_i^t|\mathbf{z}^t\right) = w_i \cdot \min_j |\mathbf{z}^t - \mathbf{z}_j|, \; \mathbf{z}_j \in D\psi_i$$

where $w_i \in \mathbf{w}$ is the weight of source i and $|\mathbf{z}^t - \mathbf{z}_j|$ computes the Euclidean distance between query point $\mathbf{z}^t$ and each data point $\mathbf{z}_j \in D_{\psi_i}$ contributed by source i.

IV. EXPERIMENTAL SETUP

Here we present the details of a first implementation of the DWL algorithm, within a simulated robot driving domain.

A. *Racetrack Driving Domain*

Empirical validation of the DWL algorithm is performed within a simulated robot driving domain, reflecting our focus on motion control policies. The robot is tasked with driving along a racetrack while staying within the track bounds. Task execution ends when the robot either completes the track (10.34 m length) or drives out of bounds.

Robot sensing within the domain consists of three range-finders (oriented forward, left, and right of the robot body). Motion dynamics are governed by robot heading and speed. The system runs at 30Hz, and changes in both heading and speed are bounded ($0.6m/s$ and $0.52rad$, respectively). Gaussian noise (5%) is added to observed range information, robot heading and executed speed.

During policy execution, the observations computed by the robot are 5-dimensional: left sensor range (lr), center sensor range (cr), right sensor range (rr), a ratio of the right and left ranges (lr/rr) and a flag indicating which track border (left or right) is observed by the center sensor. This final dimension is binary; all other dimensions are continuous-valued. The actions predicted by the robot are 2-dimensional and both continuous-valued: target speed and change in heading. Teacher demonstrations are performed via teleoperation, with the teacher commands indicating to (in/de)crease speed or (in/de)crease heading.

B. *Data Sources*

Within this domain, three data sources are available:

1) Teacher demonstration (Source T)
2) Policy execution by the learner (Source P)
3) Advice-modified executions (Source A)

Figure 1 presents these sources.

Data from source T is produced by human *teacher teleoperation* of the simulated robot during task execution. Source P is produced from *learner policy* executions of the task, by having the human teacher indicate well performing subsets K of a learner execution. The recorded execution data from the subsets $\{\mathbf{z}^t, \mathbf{a}^t\} \in K$ is then added as demonstration data to the source P dataset D_{ψ_P}.

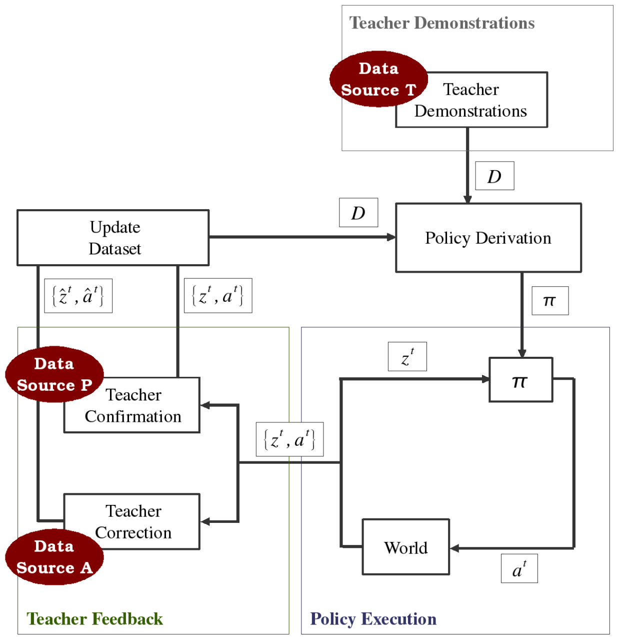


Fig. 1. Data sources for the experimental domain.

Source A is produced from *advice-modified* learner policy executions. Here the teacher indicates poorly performing subsets K of the execution, as well as a corrective advice-operator. *Advice-operators* perform simple mathematical computations that produce continuous-valued corrections for data point observations or actions. Applied to a subset of the learner execution trace, they result in synthesized, corrected, data. Three advice-operators were employed for this domain: *Turn* (less/more/none), *Change Speed* (less/more/none) and *Curve* (less/more). For a more complete discussion of advice-operators, we refer the reader to [3].

C. *Evaluation Metrics*

The evaluation metrics considered within this domain are execution success and efficiency. *Success* is measured by the length of track traversed (m). *Efficiency* is measured by mean executed speed (m/s). The weight update in DWL requires that the system provide rewards for learner executions. Reward in our system is formulated as a simple combination of success and efficiency.

During learner practice, source selection weights are continuously updated as new data is provided from various sources. A final source weighting $\mathbf{w}_f$ and dataset $\mathbf{D}_f$ are the result. For evaluation purposes, a variety of policies are further derived from $\mathbf{D}_f$, as defined in Table IV-C.

Data	Weight	Policy Name
$\mathbf{D}_f$	$\mathbf{w}_f$	*All Learned*
$\mathbf{D}_f$	equal	*All Equal*
$D_{\psi_T} \in \mathbf{D}_f$	-	*Source T*
$D_{\psi_P} \in \mathbf{D}_f$	-	*Source P*
$D_{\psi_A} \in \mathbf{D}_f$	-	*Source A*

TABLE I. Evaluation Policies

V. Empirical Results and Discussion

This section presents empirical results from the DWL implementation. Data sources are demonstrated to be unequally reliable in this domain. The automatic data source weighting under DWL is shown to outperform an equal source weighting scheme, and furthermore to perform as well as the best contributing expert.

A. Unequal Data Source Reliability

To explore the reliability of each data source, track executions were performed using policies derived exclusively from one source (policies *Source T*, *Source P* and *Source A*). The results of these executions are presented in Figure 2 (solid bars). The performances of the three sources differ in both efficiency and success, confirming that in this domain the multiple data sources are indeed not equally reliable.

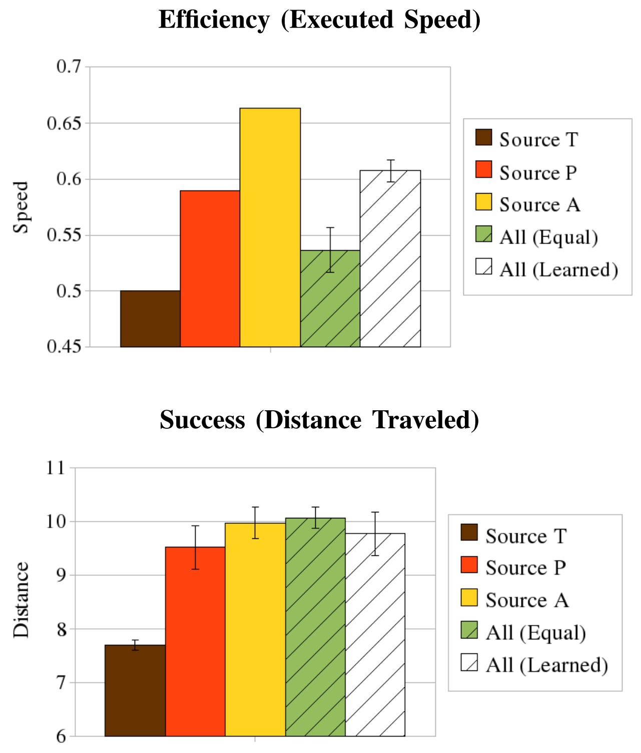


Fig. 2. Mean execution speeds and distances traveled during test executions (mean of 20 executions, 1-standard deviation error bars). Executions with exclusively one data source (solid bars) are compared to executions using all data sources (hashed bars).

B. Performance Improvement with Weighting

To examine the effects of data source weighting, executions were first performed using a policy with equal source weights (policy *All Equal*). Figure 2 shows the performance of this policy to match the best expert in success, but underperform two of the three experts, as well as the average of all three ($0.54 \pm 0.02\frac{m}{s}$ compared to $0.58 \pm 0.07\frac{m}{s}$), in efficiency (green hashed bars).

By contrast, source weights learned under the DWL algorithm (policy *All Learned*) were able to outperform the average expert efficiency. The average performance over 20 test executions with the learned weight $\mathbf{w}_f$ is shown in Figure 2 (white hashed bars). In execution speed, the learned weighting displays superior performance over the equal weighting ($0.61 \pm 0.02\frac{m}{s}$ compared to $0.54 \pm 0.02\frac{m}{s}$). In distance traveled, similar behavior is seen, with both giving near perfect performance. Furthermore, the performance of this policy begins to approach that of the most reliable data source. The DWL algorithm thus is able to combine information from *all* of the data sources in such a manner as to outperform or match the performances of most contributing experts, and to approach the performance of the best expert.

C. Automatically Learned Source Weights

The selection weights learned iteratively during practice, and that result in $\mathbf{w}_f$, are presented in Figure 3 (solid lines). These weights appropriately come to favor Source A (Fig. 2). Recall that since expert selection probability depends also on data support, however, this learned weighting scheme in theory still allows for the selection of sources other than A. For reference, also shown is the fractional source composition of D, which changes as practice incorporates new data from various sources (dashed lines). Note that not all data sources are available at the start of practice, since some (sources P and A) produce demonstration data by building on learner executions.

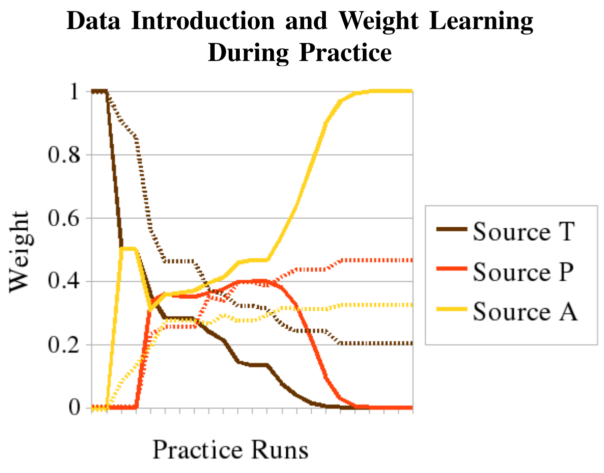
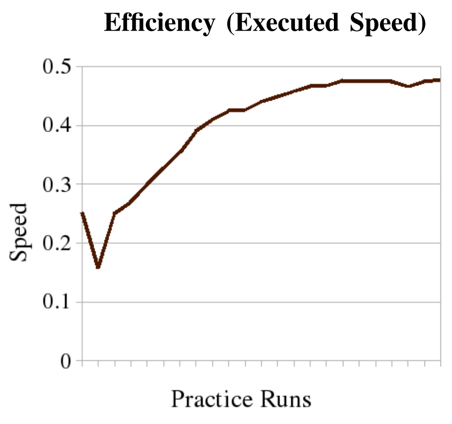


Fig. 3. Data source weight learning during the practice runs (solid lines). For reference, the fractional contribution (in number of data points) of each data source to the full data set D is additionally provided (dashed lines).

The mean execution speed and distance traveled along the track during the practice runs are presented in Figure 4. Both measures are shown to improve as a result of learner practice, and thus with the introduction and weighting of new data.

D. Discussion

DWL requires a single reward for an entire execution, and not at every execution timestep. The reward therefore only needs to evaluate overall performance, and be sufficiently



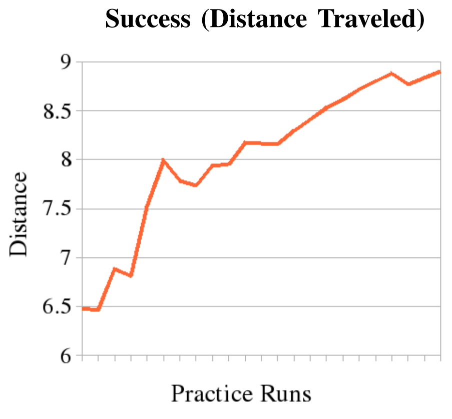


Fig. 4. Mean execution speed and distance traveled during the practice runs (running average, 23 practice runs).

rich to learn the data source *weights*; it is not necessary that it be sufficiently rich to learn the *task*.

The experimental domain presented here provides reward as a function of performance metrics unrelated to world state. The execution reward, therefore, *is not* a state reward. In this case, the DWL reward distribution assumes each *expert* to have contributed to the performance in direct proportion to the number of actions they recommended. By contrast, if the execution reward *is* a state reward, then the distribution formulation of DWL assigns equal reward to each *state* encountered during the execution. In this case, a more sophisticated approach to reward back-propagation would likely improve this algorithm.

Expert selection probabilities under DWL depend on data support and overall task performance. An interesting extension to the algorithm could further anchor task performance measures to state, and thus consider the *varying* performance abilities of experts in different areas of the state-space.

To conclude, data sources are shown to be unequally reliable within this domain (Fig. 2). Important to remember is that the learner is *not* able to choose the source from which it receives data. Furthermore, even a poor data source can be useful, if it is the only source providing data in certain areas of the state-space. DWL addresses both of these concerns. Though the learner is not able to choose the data source, it is able to favor data from certain sources through the DWL weighting scheme. Furthermore, when selecting a data source at prediction time, both this weight *and* the state-space support of the data source are considered.

VI. CONCLUSION

This work explores the incorporation of multiple data sources within a *Learning from Demonstration* paradigm. In particular, we consider *reliability* differences between multiple demonstration data sources. *Demonstration Weight Learning (DWL)* is introduced as a Learning from Demonstration algorithm that explicitly reasons about multiple data sources, and through a weighting scheme leverages their reliability. Data source weights are automatically determined, and dynamically updated according to an adaptive *expert learning* inspired approach. A first implementation of DWL is presented within a simulated robot driving domain. Data sources are found to differ in task execution reliability, and data source weighting is shown to impact task performance in both success and efficiency. Furthermore, DWL is shown to approach the performance of the best data source expert, and to outperform an equal source weighting scheme.

REFERENCES

[1] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Proceedings of NIPS'07*, 2007.
[2] B. Argall, B. Browning, and M. Veloso. Learning to select state machines on an autonomous robot using expert advice. In *Proceedings of ICRA '07*, 2007.
[3] B. Argall, B. Browning, and M. Veloso. Learning robot motion control with demonstration and advice-operators. In *Proceedings of IROS '08*, 2008.
[4] C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning for control. *Artificial Intelligence Review*, 11:75–113, 1997.
[5] C. G. Atkeson and S. Schaal. Robot learning from demonstration. In J. Douglas H. Fisher, editor, *Proceedings of ICML '97*, 1997.
[6] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multiarm bandit problem. In *36th Annual Symposium on Foundations of Computer Science*, 1995.
[7] D. C. Bentivegna. *Learning from Observation Using Primitives*. PhD thesis, College of Computing, Georgia Institute of Technology, 2004.
[8] M. Bowling and M. Veloso. Convergence of gradient dynamics with a variable learning rate. In *Proceedings of ICML '01*, 2001.
[9] S. Calinon and A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of HRI '07*, 2007.
[10] S. Chernova and M. Veloso. Confidence-based learning from demonstration using Gaussian Mixture Models. In *Proceedings of AAMAS '07*, 2007.
[11] 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.
[12] M. N. Nicolescu and M. J. Mataric. Methods for robot task learning: Demonstrations, generalization and practice. In *Proceedings of AAMAS '03*, 2003.
[13] E. Oliveira and L. Nunes. *Learning by exchanging Advice*. Springer, 2004.
[14] P. K. Pook and D. H. Ballard. Recognizing teleoperated manipulations. In *Proceedings of ICRA '93*, 1993.
[15] H. Robbins. Some aspects of the sequential design of experiments. *Bulletin American Mathematical Society*, 55:527–535, 1952.