2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

Learning Robot Motion Control with Demonstration and Advice-Operators

Brenna D. Argall, Brett Browning and Manuela Veloso

***Abstract*— As robots become more commonplace within society, the need for tools to enable non-robotics-experts to develop control algorithms, or *policies*, will increase. Learning from Demonstration (LfD) offers one promising approach, where the robot learns a policy from teacher task executions. Our interests lie with robot *motion* control policies which map world observations to continuous low-level actions. In this work, we introduce *Advice-Operator Policy Improvement* (A-OPI) as a novel approach for improving policies within LfD. Two distinguishing characteristics of the A-OPI algorithm are *data source* and *continuous state-action space*. Within LfD, more example data can improve a policy. In A-OPI, new data is *synthesized* from a student execution and teacher advice. By contrast, typical demonstration approaches provide the learner with exclusively teacher executions. A-OPI is effective within continuous state-action spaces because high level human advice is *translated* into continuous-valued corrections on the student execution. This work presents a first implementation of the A-OPI algorithm, validated on a Segway RMP robot performing a spatial positioning task. A-OPI is found to improve task performance, both in success and accuracy. Furthermore, performance is shown to be similar or superior to the typical exclusively teacher demonstrations approach.**

I. INTRODUCTION

The presence of robots within society is becoming more prevalent. Whether an exploration rover in space or recreational robot for the home, successful autonomous robot operation requires a control algorithm, or *policy*, which maps observations of the world to actions available on the robot. Policy development is currently a complex process restricted to experts within the field. However, as robots become more commonplace, the need for policy development which is straightforward and feasible for non-experts will increase.

Traditional approaches to policy development model world dynamics, and derive a mathematically-based policy. Though theoretically well-founded, these approaches depend heavily upon the accuracy of the world model. Not only does the model require considerable expertise to develop, but approximations such as linearization are often introduced for computational tractability, thereby degrading performance.

An alternative approach is to have a robot learn its control policy. One specific learning approach which has found success is *Learning from Demonstration* (LfD), e.g. [9], [10]. Within LfD, a teacher provides example executions of the task. Executions are recorded as observations of the world and selected actions. From this dataset, the learner then generalizes a control policy which maps observations to actions. Demonstration has the attractive feature of being an intuitive medium for human communication, as well as focusing the dataset to areas of the state-space actually encountered during task execution. Since it does not require expert knowledge of the system dynamics, demonstration also opens policy development to non-robotics-experts.

A desirable feature in any learning system is the ability to improve a policy based upon learner experience. Though not an explicit part of classical LfD, many LfD systems do take policy improvement steps. The most common approach is to add more teacher demonstration data in problem areas of the state space, and then re-derive the policy [6], [7].

In this work, we introduce *Advice-Operator Policy Improvement* (A-OPI) as an approach for policy improvement within an LfD framework. A key novel feature of our approach is that more data is not provided by further teacher demonstrations. Rather, new data is synthesized from *learner executions* and *teacher advice*. The new data results from the learner applying this advice to its execution data points. The synthesized data is then added to the demonstration set.

Our work focuses specifically on robot motion control within continuous state/action spaces. Correcting the state-action mapping of a policy involves indicating the correct action or state. For continuous state/action spaces, this requires providing a continuous-valued correction. Expecting the human teacher, however, to know the appropriate continuous *value* to correct these data points is neither reasonable nor efficient. To circumvent this, we have developed *advice-operators* as a language through which the human teacher provides advice to the robot student. Advice-operators perform mathematical computations on continuous-valued data points. We introduce a finite list of advice-operators, from which the human selects to provide advice. The robot learner then applies the operator to its execution data point. In this manner, we enable a framework in which advice-giving is reasonable for a human to perform, and does result in a continuous-valued correction.

We have implemented A-OPI on a Segway RMP robot performing a spatial positioning task. We show that A-OPI enables similar or superior performance when compared to a policy derived from the typical exclusively teacher demonstrations approach. Furthermore, by concentrating new data exclusively to the areas visited by the robot and needing improvement, A-OPI produces noticeably smaller datasets.

This research was sponsored 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.

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

In the next Section, we motivate the development of A-OPI, grounded within a review of related literature. Section III presents the A-OPI algorithm in depth. Section IV details our experimental implementation on a Segway RMP robot. Empirical results are presented within Section V, along with further discussion. In Section VI we conclude.

II. Motivation and Related Work

The problem of learning a mapping between world observations and proper action selection lies at the heart of many robotics applications. 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 policy $\pi : Z \rightarrow A$ selects actions based on observations of the world state. Learning for the robot consists of changing its control policy π so as to improve task performance.

A. Learning from Demonstration

One technique for robot policy learning is to learn from demonstration. Formally, a teacher demonstration $d_j \in D$ is represented as t_j pairs of observations and actions such that $d_j = \{(\mathbf{z}_j^i, \mathbf{a}_j^i)\} \in D, \mathbf{z}_j^i \in Z, \mathbf{a}_j^i \in A, i = 0 \cdots t_j$. The set D of these demonstrations are provided to the learner.

When gathering teacher demonstrations, three key decisions to make are the choices of demonstration teacher, demonstration platform, and recording sensors. These decisions heavily influences the introduction of *correspondence issues*, where demonstrations do not immediately transfer to the robot due to sensing or motion differences between the teacher and learner.

Most LfD work to date uses human demonstrators [3]. We similarly restrict the scope of our work to humans, though the algorithm itself is general to any teacher. Many approaches exist for executing and recording teacher demonstrations [7], [9]. We teleoperate our robot while recording from its own sensors, as this minimizes correspondence issues.

There are three core approaches to policy derivation from demonstration data. In the first approach, the data is used to directly approximate the underlying function mapping observations to actions [5]. In the second approach, the data is used to determine the world dynamics model $T(s'|s,a)$ and possibly a function $R(s)$ associating reward with world state [4]. In the third approach, a sequence of actions are produced by a planner after learning a model of action pre- and post-conditions [10]. Our work explores policy derivation and improvement within the first approach.

B. Policy Improvement within Learning from Demonstration

An attractive feature for any learning system is the ability to update the policy based on learner executions. Though not a part of the classical LfD formulation, a variety of LfD systems do implement policy improvement steps. For example, when a policy is derived under the world dynamics model and reward approach, execution experience may update $T(s'|s,a)$ [1], or reward-determined state values [11].

When a policy is derived by approximating the underlying mapping function, which is the approach we explore, a common technique for improvement is to provide more demonstration data. Algorithms driven by learner requests for more data [7], [8], and teacher initiation of further demonstration [6], have been shown to improve policy performance.

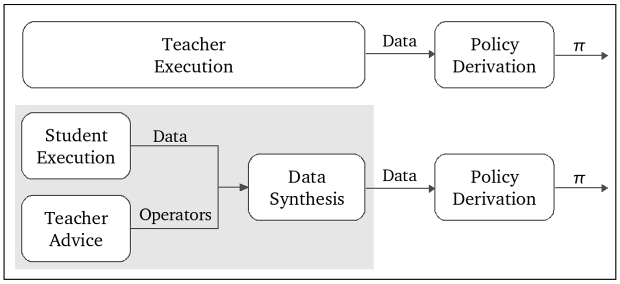

Fig. 1. Generating demonstration data. Typical approaches provide demonstration data from teacher executions (top). Our approach introduces a novel technique for generating data based upon learner executions and teacher advice (bottom, shaded box).

We contribute with this work a novel approach which instead derives new data from *learner* executions modified by advice from a human teacher (Fig. 1). Furthermore, this advice differs from the traditional Reinforcement Learning reward by providing a *correction* on the executed state-action mapping. By contrast, reward provides only an indication of the desirability of visiting a particular state; to determine the correction (i.e. the more desirable state) alternate states must be visited. This can be unfocused and intractable to optimize when working on real robot systems with an infinite number of world state-action combinations.

C. An Alternative to Teacher Demonstration

Instead of providing more teacher demonstrations, with A-OPI we *synthesize* new demonstration data based upon student executions and teacher advice. The following considerations motivate our interest in alternate data sources to teacher demonstration:

- *No need to recreate state.* This is especially useful if the world states where demonstration is needed are dangerous (e.g. lead to a collision), or difficult to access (e.g. in the middle of a motion trajectory).
- *When unable to demonstrate.* Further demonstration may actually be impossible (e.g. rover teleoperation over a 40 minute Earth-Mars communications lag).
- *Not limited by demonstrator.* A demonstrated training set is inherently limited by the demonstrator's performance, which may be suboptimal.

III. Advice-Operator Policy Improvement

We now detail the A-OPI algorithm. The approach consists of two phases. During the *demonstration phase*, a set of teacher demonstrations is provided to the learner. From this the learner generalizes an initial policy. During the *advising phase*, the learner executes this initial policy. Advice on the learner execution is offered by a human advisor, and is used by the learner to update its policy. Psuedo-code for the A-OPI algorithm is provided in Figure 2. To begin, we discuss the central technique of our advising approach: advice-operators.

A. Advice-Operators: How Humans Provide Robot Advice

The purpose of advice is to correct the robot's policy. Though this policy is unknown to the human advisor, it is represented by observation-action mapping pairs. To correct the policy, our approach therefore offers corrective information about observation-action pairings from a learner execution. However, as previously mentioned, to have a human provide continuous-valued corrective information represents a significant challenge.

We contribute a novel technique to address this issue by providing corrective information through *advice-operators*. Key characteristics of advice-operators are that they:

- Perform mathematical computations on data points.
- Are defined commonly between the student and advisor.
- May be applied to observations or actions.

We concretely define an advice-operator as a mathematical computation performed on an observation input or action output. An example of an observation-modifying operator is "reset goal to $\tilde{x}$," so that an observation previously computed with the goal x is recomputed using goal $\tilde{x}$. An example of an action-modifying operator is "increase speed," so that the executed value of the speed action is increased.

A key insight to the A-OPI approach is that pairing a modified observation (or action) with the *executed* action (or observation) now represents a corrected mapping. Assuming accurate policy derivation techniques, adding this data point to the demonstration set and re-deriving the policy will thus also correct the policy.

B. Demonstration Phase

In the first, demonstration phase of the algorithm, the learner derives an initial policy from the demonstration set D. This set is populated with data recorded during teacher executions of the task. The learner generalizes from this data to build an approximation to the observation-action mappings it contains. An initial policy π is the result (Fig. 2, line 01).

C. Advising Phase

In the second, advising phase of the algorithm, the learner policy is improved. The learner executes its current policy. The teacher observes this execution, and offers improvement advice. This advice is interpreted by the learner to modify its executed data points. The interpretation happens through advice-operators, defined within the set $\mathbb{OP}$. The modified data points are added to the demonstration dataset and a new, improved policy is derived.

For each advising run, an observed goal state $z^{goal} \in Z$ is provided (line 02). To begin the robot performs the task (*Execute*, lines 04-08), and the advisor then provides advice on this performance (*Advise*, lines 09-17).

First, the learner executes the task, producing execution trace d. The learner executes until achieving goal state z^{goal}. At each timestep the learner selects action $\mathbf{a}^t$ according to $\pi(\mathbf{z}^t)$ (line 06). This action is recorded, along with the observation $\mathbf{z}^t$, within the execution trace d (line 07).

```
00 Given $D$, $Z$, $\mathbb{OP}$
01   $\pi \leftarrow$ policyDerivation$(D)$
02   Given $\mathbf{z}^{goal} \in Z$
03     $d \leftarrow \{\}$
04     Execute
05       while $\mathbf{z}^t \; != \mathbf{z}^{goal}$
06         $\mathbf{a}^t \leftarrow \pi(\mathbf{z}^t)$
07         $d \leftarrow \{d, \; (\mathbf{z}^t, \mathbf{a}^t)\}$
08       end
09     Advise
10       $\{op, \hat{d}\} \leftarrow$ advice$(d)$ , $op \in \mathbb{OP}, \hat{d} \in d$
11       foreach $(\mathbf{z}^k, \mathbf{a}^k) \in \hat{d}$
12         if observation-modifying
13           $(\mathbf{z}^k, \mathbf{a}^k) \leftarrow (op(\mathbf{z}^k), \mathbf{a}^k)$
14         elseif action-modifying
15           $(\mathbf{z}^k, \mathbf{a}^k) \leftarrow (\mathbf{z}^k, op(\mathbf{a}^k))$
16         end
17       end
18     Update
19       $D \leftarrow \{D, \; \hat{d}\}$
20       $\pi \leftarrow$ policyDerivation$(D)$
```

Fig. 2. Psuedo-code for the A-OPI algorithm.

Second, the advisor provides advice on the execution d. The advisor indicates an advice-operator $op \in \mathbb{OP}$ and a subset of execution points $\hat{d} \in d$ over which to apply the operator (line 10). The operator is applied to each execution point $(\mathbf{z}^k, \mathbf{a}^k)$ in the subset $\hat{d}$ (lines 11-17). The data point is modified according to line 13 or 15 depending upon whether *op* is observation- or action-modifying, respectively.

Finally, the set $\hat{d}$ of modified data points is added to the dataset D (line 19). A new policy π is then derived from this set (line 20).

D. A-OPI for Robot Motion Control

A-OPI is targeted for low level robot motion control. The algorithm learns a mapping from observations of world state to continuous-valued robot actions. Here we overview *regression* as the tool for deriving continuous-valued policies from demonstration data, and the *advice-operators* we have developed as tools to modify motion control policies.

1) Policy Derivation with Regression: Regression techniques predict actions using a current observation and the training (demonstration) dataset. A wealth of regression approaches exist, and we emphasize that any may be used with A-OPI. Our specific implementation employs a form of Locally Weighted Learning [2]. Given current observation $\mathbf{z}^t$, action $\mathbf{a}^t$ is predicted through an averaging of data points in D, weighted by their kernelized distance to $\mathbf{z}^t$. Thus,

$$\mathbf{a}^t = \sum_{(\mathbf{z}_i, \mathbf{a}_i) \in D} w_i^t \cdot \mathbf{a}_i, \quad w_i^t = e^{\left(\mathbf{z}_i - \mathbf{z}^t\right)\Sigma^{-1}\left(\mathbf{z}_i - \mathbf{z}^t\right)^T} \qquad (1)$$

where the weights w_i^t have been normalized over i, and Σ^{-1} is a constant parameter scaling observation dimension (tuned through cross-validation). Regression tuning is external to A-OPI and tied to policy derivation, occurring for initial derivation (line 01) and possibly for re-derivations (line 20). In our implementation the distance computation is Euclidean, and the kernel is Gaussian.

2) Motion Control Advice-Operators: The advice-operators which we have developed for motion control are presented within Figure 3. These operators were developed with the aim of representing corrections which would be straightforward and intuitive for a human to identify. To select an operator, therefore, the human need only rely upon his own observation of the robot execution, without requiring further specifics; for example, exact speed value. Note that many operators do not take any parameters, and those which do require only a binary indication.

	Operator Description	Parameter
0	Reset goal, compute observation	
1	No turning	
2	Start turning	[cw/ccw]
3	Smooth rotational speed	[ac/de]
4	Turn [less/more] tightly	[less/more]
5	No translation	
6	Smooth translational speed	[ac/de]
7	Translational [ac/de]celeration	[ac/de]
8	Stop all motion	

Fig. 3. Advice-operators for the spatial positioning task. [Key: cw=clockwise, ccw=counterclockwise, ac=accelerate, de=decelerate]

To illustrate, we consider the "Translational [ac/de]celeration" operator as an example. Suppose the teacher indicates the "accelerate" parameter and a chunk of 10 data points over which to apply the operator. Our implementation of this operator functions by augmenting the actions by linearly increasing percentages of the executed speed; for example updating the translational speed of point 0 to $a^0 \leftarrow 1.1 \cdot a^0$, point 1 to $a^1 \leftarrow 1.2 \cdot a^1$, and so forth through to the final point 9 to $a^9 \leftarrow 2.0 \cdot a^9$.

In practice, advice may be provided during or after the robot execution (but in either case will influence behavior only *after* a policy update). In our advising-interface implementation, a graphical representation of the 2-D path taken by the robot is provided post-execution. This is a tool through which the human flags recorded observation-action pairings for modification, by selecting segments of the displayed path.

IV. EXPERIMENTAL DESIGN

In this section we present our experimental setup, including the strategy for policy development and task evaluation.

A. Task: Spatial Positioning with Heading

The task chosen to experimentally validate the A-OPI algorithm is spatial positioning with a Segway RMP robot (Fig. 4). The spatial positioning task consists of attaining a 2D planar target position (x_g, y_g), with a target heading θ_g.

The Segway RMP is a dynamically balancing differential drive robot produced by Segway LLC. The platform accepts wheel speed commands, but does not allow access to its balancing control mechanisms. The inverted pendulum dynamics of the robot present an additional element of uncertainty for low level motion control. Furthermore, for this task smoothly coupled rotational and translational speeds were preferred, in contrast to turning on spot to θ_g after attaining (x_g, y_g). To mathematically define such trajectories for this specific robot platform is thus non-trivial, encouraging the use of alternate control approaches such as A-OPI. That the task is straightforward for a human to evaluate and correct further supports A-OPI as a candidate approach. While the task was chosen for its suitability to validate A-OPI, to our knowledge this work also constitutes the first implementation of such a motion task on a real Segway RMP platform.

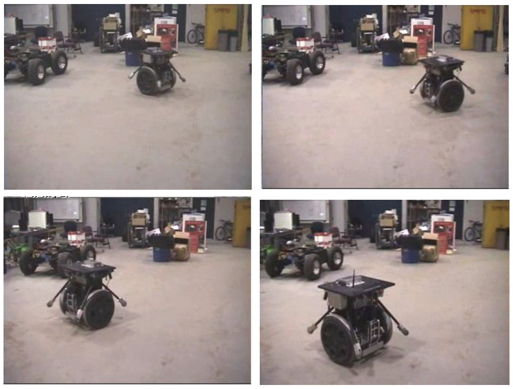

Fig. 4. Segway RMP robot performing the spatial positioning task.

The observations and actions for this task are 3- and 2-dimensional, respectively. Let the current robot position and heading within the world be represented as (x_r, y_r, θ_r), and the vector pointing from the robot position to the goal position be $(x_v, y_v) = (x_g - x_r, y_g - y_r)$. An observation consists of: squared Euclidean distance to the goal $(x_v^2 + y_v^2)$, the angle between the vector (x_v, y_v) and robot heading θ_r, and the difference between the current and target robot headings $(\theta_g - \theta_r)$. An action consists of: translational and rotational speeds. The robot samples these values from wheel encoders at 30 Hz.

B. Policy Development

The set D is seeded with demonstrations recorded as the teacher teleoperates the robot learner (here 9 demonstrations, 900 data points). We will refer to the initial policy derived from this dataset as the *Baseline Policy*.

Policy improvement proceeds as follows. A goal is selected (without replacement) from a practice set consisting of (x, y, θ) goals drawn uniformly within the bounds of the demonstration dataset $([-0.33, 4.5]m, [-4.0, 0.17]m, [-3.1, 1.1]rad)$. The robot executes its current policy to attain this goal. The advisor observes this execution. If the execution is considered poor, the advisor offers policy improvement information. The policy is re-derived. Drawing a new goal then initiates another practice cycle.

Three policies were developed using distinct techniques, differing in *what* was offered as policy improvement information. The first provided advice exclusively, in the form of advice-operators (*A-OPI Advised Policy*). The second involved an initial phase of exclusively more teleoperation, followed by a phase of exclusively offering advice (*A-OPI Hybrid Policy*). The third provided further teleoperation teacher demonstrations exclusively (*Teleoperation Policy*). We refer to these collectively as the *improvement policies*.

C. Policy Evaluation

Policy performance is evaluated on a test set, consisting of 25 (x, y, θ) goals, again drawn from a uniform distribution within the bounds of the demonstration dataset. The test set is independent, and no executions associated with it receive policy improvement information.

Policies are evaluated for accuracy and success. *Accuracy* is defined as Euclidean distance between the final robot and goal positions $e_{x,y} = \|(x_g - x_r, y_g - y_r)\|$, and the final robot and goal headings $e_\theta = |\theta_g - \theta_r|$. We define *success* generously as $e_{x,y} < 1.0\ m$ and $e_\theta < \frac{\pi}{2}\ rad$.

V. Empirical Results and Discussion

Policy performance improved with A-OPI advising, in both execution success and accuracy. When compared to the approach of providing more teleoperation data, final improvement amounts were found to be similar or superior. Furthermore, this performance was achieved with a similar number of practice executions, but smaller final dataset D.

For each policy improvement approach, the policy improvement phase was halted once performance on the test set no longer improved. The final A-OPI Advised, A-OPI Hybrid and Teleoperation Policies contained data from 69, 68 and 60 executions, respectively (with the first 9 demonstrations for each attributable to seeding with the Baseline Policy).

A. Increase in Successful Executions

Figure 5 presents the percent execution success of each policy on the independent test set. When compared to the Baseline Policy, all policy improvement approaches display increased success. Both the advised A-OPI policies additionally achieve higher success than the Teleoperation Policy.

Test Set Percent Success

A-OPI Advised	A-OPI Hybrid	Teleop	Baseline
88%	92%	80%	32%

Fig. 5. Percent successfully attained test set goals.

B. Improved Accuracy

Figure 6 plots the average position and heading error on the test set goals, for each policy. For positional error, all improvement policies display similar performance, which is a dramatic improvement over the Baseline Policy. For heading, A-OPI Advised reduces more error than A-OPI Hybrid, with both showing marked improvements over the Baseline Policy. By contrast, the Teleoperation Policy displays *no* overall improvement in heading error.

That heading error proved in general more difficult to improve than positional error is consistent with our prior experience with this robot platform, being highly sensitivity to rotational dead reckoning error accumulation.

Test Set Error, Final Policies

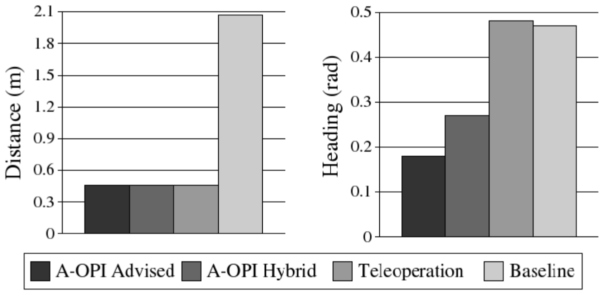

Fig. 6. Average test set error on target position (left) and heading (right), with the *final* policies.

Intermediate policies are produced as a result of our iterative policy development approach (Section IV-B). A sampling of these policies were also evaluated on the test set, to mark improvement progress for each policy improvement technique. Figure 7 shows the average position and heading error, on the test set goals, for these intermediate policies.

Test Set Error, Intermediate Policies

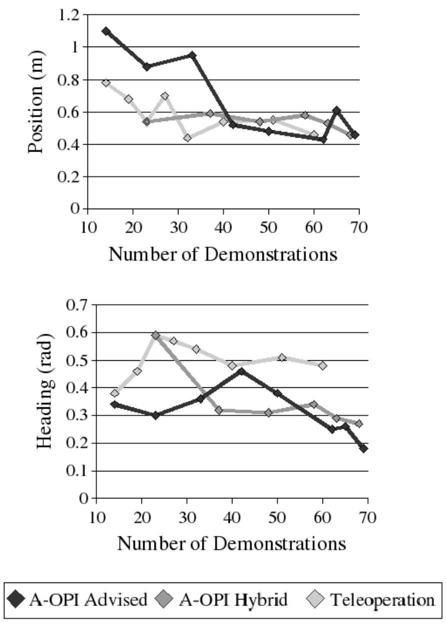

Fig. 7. Average test set error on target position (top) and heading (bottom), with *intermediate* policies (shown against the number of advised and/or teleoperated demonstrations).

Superior heading error is consistently produced by advising, throughout policy improvement. Greater positional error improvement is initially seen through teleoperation. Advising reduces positional error more slowly, but does eventually converge to the teleoperation level.

The teleoperation phase of the A-OPI Hybrid Policy resulted from seeding with an intermediate Teleoperation Policy (23 demonstrations). Following this seeding, advising occurs. The A-OPI Hybrid Policy thus initially displays the superior reduction in positional error, and inferior reduction in heading error, of the Teleoperation Policy, followed by substantial reductions in heading error through advising.

C. More Focused Improvement

The results of Figure 7 are plotted against the number of executions contributing to the set D. How many *data points* are added with each execution, however, varies greatly depending upon whether the execution is advised or teleoperated (Fig. 8). This is because, in contrast to teleoperation, only subsets of an advised execution are added to D; in particular, only those execution points which actually receive advice. States visited during good performance portions of the student execution are *not* redundantly added to the dataset. In this manner, the final policy performances shown in Figure 6 are achieved with much *smaller* datasets for both A-OPI policies, in comparison to the Teleoperation Policy.

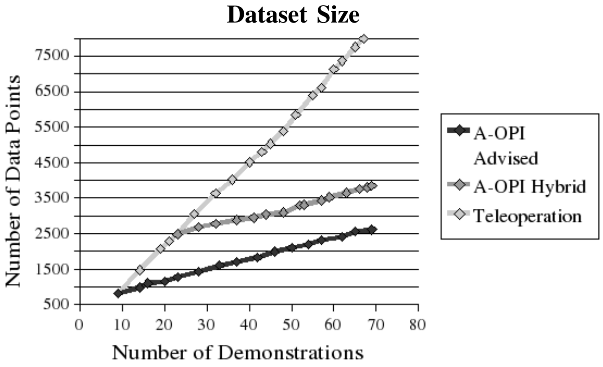

Fig. 8. Dataset size growth with demonstration number.

D. Discussion

The empirical results confirm that a human teacher was able to effectively advise within *continuous* action spaces. This occurred without the teacher providing continuous-values for the corrections, or requiring value-based execution details (e.g. speed) to select operators. The robot, and not the human, applied the operator.

The teacher was able to provide advice quickly, because the algorithm is robust to fuzzy selections of execution points. Since regression treats each data point independently, a point added to the dataset does not depend upon whether nearby points from the execution were also added. Furthermore, operators are "smart" enough to check for particular data point qualities when necessary. For example, a point which already had zero rotational speed would be skipped by the "No turning" operator.

We finish with comments on the hybrid policy. The hybrid policy was seeded with an intermediate Teleoperation Policy and then advised, in an attempt to exploit the strengths of each approach. One direction for future work could interleave providing advice and more teleoperation. Alternately, a sufficiently populated demonstration set could be provided at the start. This is fundamentally different from providing more teleoperation in response to *student* executions, however. It requires prior knowledge of all states the student will visit; generally impossible in continuous-state real worlds.

The hybrid approach was motivated from noting that (i) more teleoperation quickly reduced initial positional error but struggled to reduce heading error, and (ii) advice gradually reduced both types of error overall. The reason, we suspect, is that teleoperation is more effective for reaching previously undemonstrated areas of the state space; by contrast, advice can only be offered on states already visited, and therefore reachable with the current policy. Correspondingly, advice is more effective at correcting visited states; by contrast, it is difficult for the teacher to revisit real world states through teleoperatation, especially when they lie in the middle of a motion trajectory.

VI. CONCLUSION

We have introduced Advice-Operator Policy Improvement (A-OPI) as an algorithm for policy improvement based on human advice within a Learning from Demonstration framework. A-OPI is distinguished by improving policies within *continuous* action spaces, and providing an alternative data source to teacher demonstrations. As such, it is suitable for developing low level motion control policies. We have presented a first implementation of A-OPI on a real robot system. Policy modifications due to A-OPI were shown to improve policy performance on a Segway RMP robot, both in execution success and accuracy. Furthermore, performance was found to be similar or superior to the typical approach of providing more teacher demonstrations.

REFERENCES

[1] P. Abbeel and A. Y. Ng. Exploration and apprenticeship learning in reinforcement learning. In *Proceedings of ICML '05*, 2005.
[2] C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning for control. *Artificial Intelligence Review*, 11:75–113, 1997.
[3] C. G. Atkeson and S. Schaal. Robot learning from demonstration. In J. Douglas H. Fisher, editor, *Proceedings of ICML '97*, 1997.
[4] J. Bagnell, A. Y. Ng, and J. Schneider. Solving uncertain Markov Decision Problems. Technical report, Robotics Institute, CMU, 2001.
[5] D. C. Bentivegna. *Learning from Observation Using Primitives*. PhD thesis, College of Computing, Georgia Institute of Technology, 2004.
[6] S. Calinon and A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of HRI '07*, 2007.
[7] S. Chernova and M. Veloso. Confidence-based learning from demonstration using Gaussian Mixture Models. In *Proceedings of AAMAS '07*, 2007.
[8] D. H. Grollman and O. C. Jenkins. Dogged learning for robots. In *Proceedings of ICRA '07*, 2007.
[9] A. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings of ICRA '02*, 2002.
[10] M. N. Nicolescu and M. J. Mataric. Methods for robot task learning: Demonstrations, generalization and practice. In *Proceedings of AAMAS '03*, 2003.
[11] M. Stolle and C. G. Atkeson. Knowledge transfer using local features. In *Proceedings of ADPRL '07*, 2007.