Batch versus Interactive Learning by Demonstration

Peng Zang , Runhe Tian*, Andrea L. Thomaz and Charles L. Isbell

College of Computing, Georgia Institute of Technology, Atlanta, Georgia 30332–0259

Email: {pengzang, athomaz, isbell}@cc.gatech.edu, *runhe@gatech.edu

Abstract—**Agents that operate in human environments will need to be able to learn new skills from everyday people. Learning from demonstration (LfD) is a popular paradigm for this. Drawing from our interest in Socially Guided Machine Learning, we explore the impact of interactivity on learning from demonstration. We present findings from a study with human subjects showing people who are able to interact with the learning agent provide better demonstrations (in part) by adapting based on learner performance which results in improved learning performance. We also find that interactivity increases a sense of engagement and may encourage players to participate longer. Our exploration of interactivity sheds light on how best to obtain demonstrations for LfD applications.**

I. Introduction

Agents (*e.g.,* robotic helpers, non-player characters in games, *etc.*) that operate in human environments will be more flexible if they are able to learn new tasks and skills from their end-users. We recognize that while the average person is not familiar with machine learning techniques, they are intimately familiar with social learning (*e.g.,* tutelage, imitation).

Socially Guided Machine Learning (SGML) [1] explores ways in which machine learning can be designed to more fully take advantage of natural human interaction and tutelage. It asks questions like "how do people want to teach agents?" and "how do we design agents to learn effectively from natural human interaction and instruction?".

Inspiration for SGML comes (in part) from Situated Learning Theory, a field of study that looks at the social world of children and how it contributes to their development. A key concept is scaffolding, where a teacher provides support such that a learner can achieve something they would not be able to accomplish independently [2], [3].

The teaching and learning process of a situated learning interaction are tightly coupled. A good instructor maintains a mental model of the learner (*e.g.,* what is understood, what remains unknown) in order to provide appropriate scaffolding to support the learner's needs. Some examples of scaffolding mechanisms are: attention direction, feedback, regulating the complexity of information, and guiding the learner's exploration. In general, this is a complex process where the teacher dynamically adjusts their support based on the learner's demonstrated skill level. The learner, in turn, helps the instructor by making their learning process transparent through communicative acts, and by demonstrating their current knowledge and mastery of the task [4], [5]. Overall, this suggests that for machine learners to be successful, we must have a tightly coupled interaction in which the learner and instructor cooperate to simplify the task for the other.

In this paper we address the impact that a tightly versus loosely coupled interaction has on the learning process. We explore this in a Learning by Demonstration (LfD) [6] context. In LfD, a control policy or plan is learned from examples, or demonstrations provided by a teacher. This can be approached with a supervised learning framework [7], by derivation of the reward function [8], or by derivation of the model [9]. Regardless of the approach, LfD is interesting to us because it aligns well with SGML; it typically does not require expert knowledge of domain dynamics nor of machine learning on the part of the teacher. It has also led to successes ranging from helicopter flying [10], to simulated robosoccer [11], to ball seeking for the Sony Aibo [12]. Our exploration of interactivity in SGML contributes to the field of LfD, providing concrete data on how best to obtain demonstrations from non-expert humans for LfD applications.

In this paper we compare two teaching paradigms: interactive LfD and batch LfD. In interactive LfD, the teacher provides a series of demonstrations interspersed with interactions with the learning agent, whereas in batch LfD, there are no interactions. Unlike sophisticated approaches to interactive LfD based on active learning principles [12]–[14], we use a relatively simple framework. Our interactivity is one way. Human teachers see and evaluate agent performance but the agent has no direct communication channel to the teacher. For example, the agent cannot ask for demonstrations. Similarly, our LfD algorithm, a variant of off-policy Q-Learning, is also relatively simple. This simplicity reflects our focus on the effect of interactivity on learning. A complex interaction framework or LfD algorithm could conflate matters.

Our experimental testbed is a game platform where players teach a computer agent to play Pac-Man. We find that:

- **Interactivity improves learning performance**. Specifically, we find that (1) interactivity improves the ability of the player to evaluate learner performance, (2) players changed their teaching strategy based on learner performance, and (3) adapted strategies result in faster learning.
- **Interactivity improves teacher experience**. Interactivity makes players feel more engaged. Specifically, they (1) felt like they made a difference and (2) felt like an active participant. Interactivity may also encourage players to participate longer.

The first empirical result is particularly interesting because it implies that the learner is able to improve its own learning environment merely through transparancy, without any direct communication channel to the teacher.
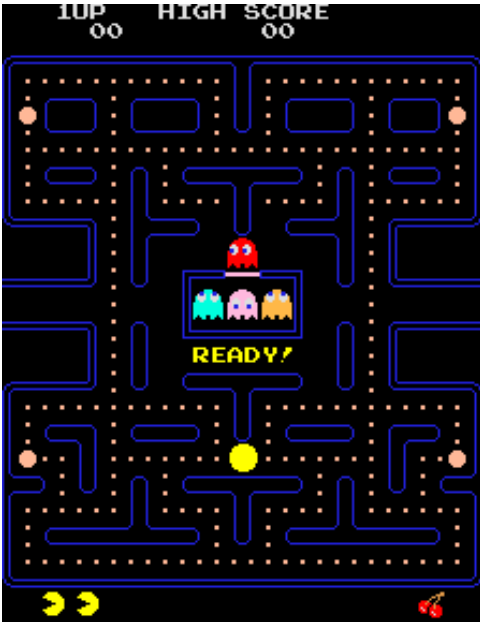
Fig. 1. Screenshot of the original arcade game, Pac-Man [15]

II. PLATFORM

In order to study how interactivity affects LfD, we ran an experiment with a Pac-Man game.

A. Domain (Pac-Man)

Pac-Man is a classic arcade game. The original version of Pac-Man (see Figure 1) is a single-player game where a human player controls the Pac-Man character around a maze. Pac-Man must avoid four ghost characters, Blinky, Pinky, Inky and Clyde, while eating dots initially distributed throughout the maze. When all dots are eaten, Pac-Man will be taken to the next level. If Pac-Man is caught by a ghost, he loses a life. When all lives (usually three) have been lost, the game ends. Near corners of the maze, there are large, flashing dots known as power dots. Eating a power dot gives Pac-Man the temporary ability to eat the ghosts. When a ghost is eaten, it returns to the ghost spawn location ("ghost jail"). In the typical version of the game, normal dots are worth 10 points, power dots are worth 50 points, and ghosts are worth 200, 400, 800, and 1600 points for the first, second, third and fourth ghosts.

Our platform uses a scaled down variant of Pac-Man. This enables our learning algorithm to learn quickly enough to be used in a real-time fashion, allowing participants to immediately see the results of their training demonstrations (in the interactive version). In our version of Pac-Man, the maze is smaller: 7 x 8. There are also fewer dots: eight normal dots and one power dot. Only one ghost, Blinky, roams the maze. Blinky is the chaser ghost, it always chases Pac-Man using Manhattan distance as its heuristic. In our scaled down variant of Pac-Man, there is only a single level. The game ends when you beat the level. Scoring is done as follows: 1000 bonus points are awarded for clearing a level (when all dots are eaten), 10 points are awarded for eating a dot (both normal and power dots), 100 points are awarded for eating the ghost, and 1000 points are deducted if Pac-Man is caught by the ghost. Maximum score is 1190 (1000 + 90 + 100).

B. Learning Algorithm (Q-Learning)

We frame Pac-Man as a Reinforcement Learning (RL) [16] problem. In particular, it is modeled as a Markov Decision Process (MDP). A MDP $(S, A, P_{s,s'}^a, R_s^a, \gamma)$ is defined by set of states S, a set of a actions A, a transition model $P_{s,s'}^a = Pr(s'|s, a)$ specifying the probability of reaching state s' by taking action a in state s, a reward model $R_s^a = r(s, a)$ specifying the immediate reward received when taking action a in state s, and the discount factor $0 \leq \gamma \leq 1$.

In our domain, state is represented by a vector containing the position of Pac-Man, the position of the ghost, nine binary variables denoting the existence of the nine dots, a jail counter denoting the remaining time the ghost must remain in its jail, and a power-mode counter representing the time remaining in power-mode. The actions are NORTH, SOUTH, EAST, and WEST. The reward function correspond to points. That is to say, eating a dot results in +10 reward, eating a ghost is +100, *etc.*

We used Q-Learning, for our learning agent. Q-Learning [17], is widely used in RL [18]–[20] and has had many positive results in playing stochastic games [21], elevator control [22] and robotics [23]. Q-Learning produces an action-value function or Q function, $Q : S \times A \to \mathbb{R}$, which maps every state-action pair to the expected utility of taking the action in the state and following the greedy policy thereafter. The greedy policy of a Q function is one that simply chooses the action with the highest Q-value for any state.

Q-Learning is an off-policy learning algorithm, meaning it can learn from example trajectories that differ from its policy. To learn from demonstrations, we take advantage of this ability and simply perform Q-Learning along the demonstrated trajectory.

We implemented a variant of the Q-Learning algorithm. For human trajectories, we used a simple replay mechanism, BTD [24], to update the Q-values. This magnifies the effect of human demonstrations and speeds up the learning process. Simple replay is not performed for self-play trajectories as they are likely less optimal and contain less accurate Q estimates.

In our implementation, Q-values are randomly initialized. We set learning rate α to 0.8, discount factor γ to 0.99, and ϵ-greedy exploration parameter ϵ to 0.9, which decays with rate 0.95 per game. Note that in the interactive learning case, ϵ is set to zero once interaction begins (after game 45). This ensures the agent performance players observe is representative of the underlying learned Q-function and not due to a random exploratory action being chosen.

C. Interface

We set up an interface for soliciting demonstrations. The interface (see Figure 2), has several components detailed in the list below and tagged in the figure as A-F. The first is the Pac-Man board (A), the second contains the demonstration controls
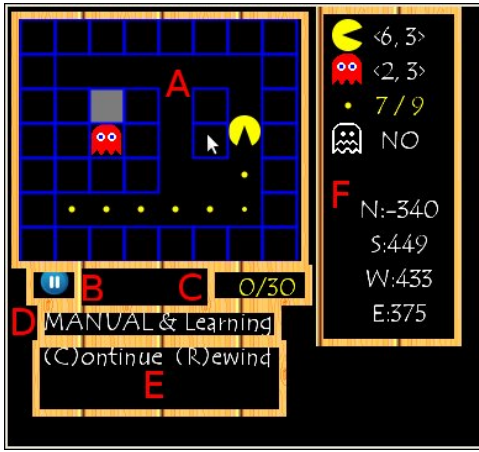
Fig. 2. Screenshot of our interface

(B, C, D, E). The right hand side (F) contains debugging information and can be ignored.

A: Pac-Man board
B: Pause/Unpause button and indicator
C: Shows number of human demonstrations stored
D: Shows the current play mode.
   "MANUAL & Learning" means the game is in player's control and the agent is updating Q-values.
   "AUTO & Learning" means agent is making moves while updating Q-values.
   "MANUAL & Playing" means player is playing the game and the agent is not learning.
E: Available control keys. 'C'ontinue is equivalent to the unpause button. Pressing 'R' rewinds the state of the game by one step. It can be repeatedly used to rewind the game to the beginning. Other control keys (not shown) are: 'P' for pausing the game, 'S' for starting a new game.
F: Debugging information. Ignored in the experiments.

III. EXPERIMENTAL DESIGN

We ran an user study with our platform to see how interactivity affects both agent learning and the teaching experience. We used a randomized, between groups study of 20 participants solicited from the campus community. Their backgrounds range from Bachelor to PhD students, as reported in an exit survey. Each of the 20 participants, did the experiment in one of two groups:

- BATCH: This group gave demonstrations directly without feedback or interaction.
- INTERACTIVE: This group gave the second half of their demonstrations with interaction.

Batch learning mode: Players were asked to demonstrate 30 consecutive games. They were allowed to rewind and correct their trajectory if they wished (*e.g.*, to correct an unintentional mistake). They received no feedback on how the agent learned. After the 30 demonstrations, the agent played 60 additional games to learn on its own. Thus, the final policy is a result of learning from these 90 trajectories.

Interactive learning mode: Players were first asked to demonstrate 15 games. The agent was then allowed to learn on its own for another 30 games (this process was performed opaquely, with all animations showing agent actions turned off). After this initial learning period, we restarted animations and allowed players to see the agent as it learned and explored on its own. *i.e.,* they were able to watch the agent controlled Pac-Man move around on the board. This continued for the remainder of the experiment. In the remaining 45 games, players were asked to watch agent play, and if they deemed necessary, provide corrective demonstrations. To provide a corrective demonstration, the player must pause the game (or wait until Pac-Man dies) and rewind play to an appropriate point from which to provide a demonstration of what Pac-Man should have done. In other words, the player plays Pac-Man to completion from the rewound point. Players were allowed a maximum of 15 corrective demonstrations.

In the study, participants were first given instructions to read. The instructions describe the rules and goal of the game. It also introduces the player to their role as a teacher, to teach the computer to play Pac-Man. Depending on which group a participant is in, we then gave specific, group appropriate instructions outlining the study progression. To motivate the participants, each participant was paid $5. To help participants understand the instructions, we provided a short demo of the important interface elements. Participants were then allowed to play a few practice games before starting the real experiment. When the participants were ready, we started the experiment. At the end of the session, an exit survey (including demographic questions) and a brief interview were conducted. Questions were mainly about teaching strategies, perceived performance of agent's learning, and comments on the teaching environment.

We logged the learned policy after each game of Pac-Man for each participant in the study. We also maintained trajectory logs for each participant. It recorded state transitions, actions, rewards and value-updates.

IV. RESULTS

A. Interactivity improves learning

To evaluate the effect of interactivity on learning, we compare logs of agent performance between interactive and batch groups. Figure 3 shows the averaged learning curves of the two groups. As the number of games (trajectories) available to the agent increases, so does its performance. Note that the composition of the games is different between the groups. The first thirty games of the batch group are human provided training games, the rest are games generated from self-play. By contrast, only the first fifteen games of the interactive group are human provided, the following thirty games are generated from self-play, and the final games (from game 45 onwards) are a mix of human demonstrations and agent self-play.

We can see that initially, batch and interactive performances are comparable. At times batch outperforms interactive, sometimes the reverse occurs. The first statistically significant difference occurs at game 63 when the interactive group
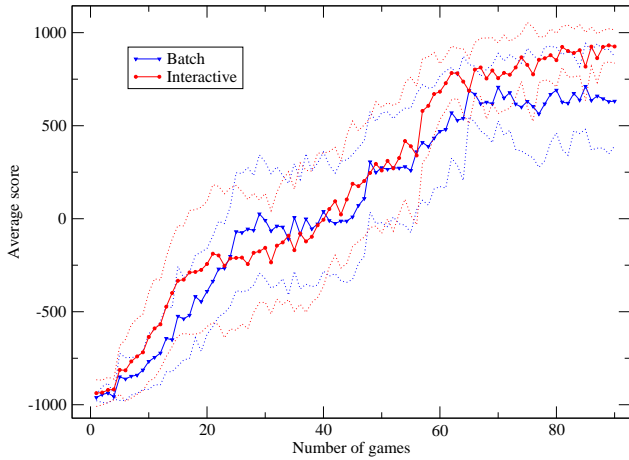
Fig. 3. Averaged learning curve for batch and interactive. Dotted curves show the lower and upper 95 percent confidence interval band.

State	Action distribution (N/S/E/W)
A	0% / 0% / 100% / 0%
B	50% / 0% / 50% / 0%
C	0% / 0% / 0% / 100%
all others	$\perp$

TABLE I
EXAMPLE PARTIAL POLICY

outperforms the batch (df=14.1, t=1.79, p=0.048). While not every game thereafter has a statistically significant difference, by the end the difference is clear: game 88 (df=11.0, t=1.96, p=0.037), game 89 (df=10.8, t=2.01, p=0.035), game 90 (df=11.45, t=2.24, p=0.023). The difference is even more evident when we consider that interactive participants only gave a total of 24 teaching demonstrations on average as compared with the 30 that batch participants gave. Players in the interactive group gave fewer demonstrations on average because they only gave corrective demonstrations (in the second half) when the agent made serious errors and the agent often began performing satisfactorily well before all corrective demonstrations were used.

To explain why interactivity improves learning performance we hypothesize the following. First, interactivity improves the ability of the player to evaluate learner performance. Second, players change their teaching strategy based on their (improved) understanding or mental model of learner behavior. Finally, the adapted strategy results in improved learning. We explore each of these hypotheses below.

1) Interactivity improves ability to evaluate the learner: In our exit interview, participants were asked to give an estimate of the performance of the final trained agent. Not all participants were willing to give an estimate, some reported that they were unsure and had no estimate to give. This occurred frequently in the batch group where half of the participants could not estimate agent behavior. By contrast, all participants in the interactive group were able to give estimates. This difference is statistically significant: df=1, $\chi^2 = 10$, p=0.0016.

Although interactive participants are better able to estimate learner performance, the estimates must also be accurate to be useful. In exit interview responses, all interactive participants estimated the final agent to perform "pretty well, though not perfect". This is borne out in empirical results. Actual performance of the final agent has a 95 percent confidence interval from a score of 836 to 1015, or alternatively, a win rate of 88 to 96 percent. This corresponds to player estimates.

2) Participants changed their teaching strategy based on learner performance: In exit interviews, 70 percent of interactive participants said they changed their teaching strategy during the session. To see if this is true empirically, we compared their teaching strategy in the first fifteen games before they have a chance to see learner performance with the strategy used in the remaining teaching demonstrations. A "strategy" is modeled as a partial policy, $\pi : S \rightarrow A$ distribution $\bigcup \perp$, mapping states to action distributions where $\perp$ indicates that the policy is not defined for the given state. We estimate players' teaching strategy from their demonstrations. This is best illustrated by an example. Suppose we see the following demonstration (trajectory) of state/action pairs: (A, East); (B, East); (C, West); (B, North) This would create the partial policy shown in Table I.

We compare two strategies or partial policies by computing the average difference between action distributions on those common states defined in both strategies. More formally the difference between two strategies is computed as $d(\pi_a, \pi_b) = \frac{1}{Z} \sum_{s \in \pi_a \bigcap \pi_b} L2E(\pi_a(s), \pi_b(s))$, where $L2E$ is the integrated square error [25] and Z is a normalization constant.

When we looked at the policy difference between the first fifteen and the later demonstrations, *i.e.,* the shift in teaching strategy, we saw a statistically significant difference for those participants who said they changed policies over those who said they did not: df=7.56, t=3.18, p=0.0065. In other words, participants who said they changed policies actually did.

To test whether the change was based (at least in part) on learner performance, we compared the amount of policy change of players with the five worst initial agents, with the amount of policy change of participants with the five best initial agents. By "initial" we mean the agent's performance after the first fifteen demonstrations, before players have a chance to interact with the learner.

A t-test shows that the players with poor initial agents changed their teaching strategy significantly more than those with good initial agents: df=6.70, t=5.97, p=0.00027. We performed linear analysis of the data (see Figure 4) and found that relative initial performance of the agent predicts the amount of policy change (r-squared=0.83). Relative performance is computed as $(score - minscore)/(maxscore - minscore)$.

3) Adapted strategy results in faster learning: To see whether adapted strategies resulted in faster learning, we first examine learning rates. We compare the learning rate of interactive participants that significantly adapted their strategy,
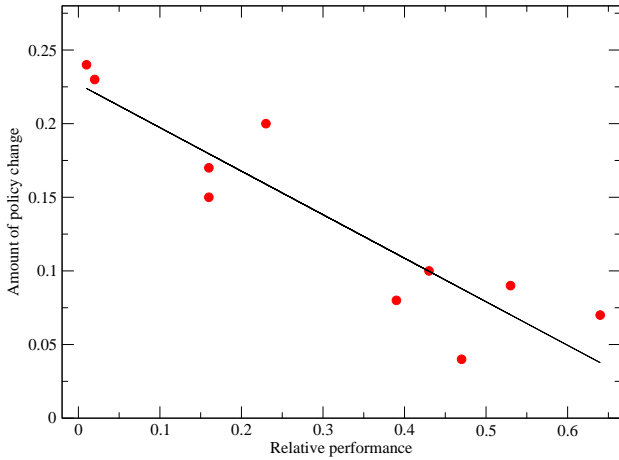
Fig. 4. Amount of policy change as a function of relative initial performance.
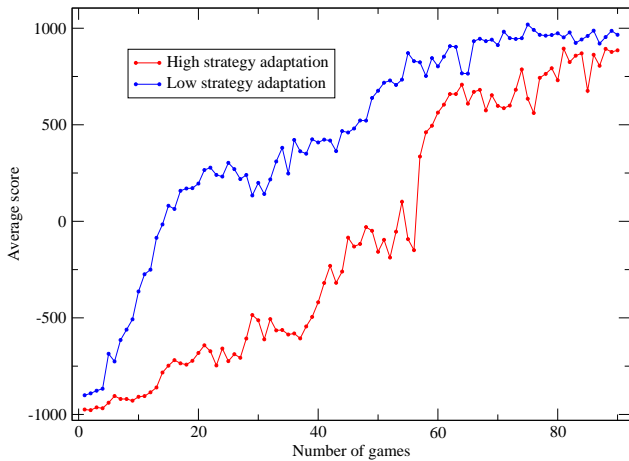

Fig. 5. Averaged learning curve of interactive participants that significantly changed their teaching strategy and those that did not.

with interactive participants that did not. Figure 5 compares the two learning curves.

For interactive participants that significantly adapted their strategy, the most rapid improvement (highest learning rate) occurred around game 60 (mean=59.7, stderr=2.9) meaning it occurred with 95% confidence somewhere in the range of game [54.0, 65.4]. This range exactly corresponds with when the additional, adapted strategies began to make up a significant portion of the teaching demonstrations. The range corresponds to [5.8, 9.2] adapted games which proportionally is [27.9%, 38.0%] of training demonstrations. We can see the resulting rapid improvement in agent performance at this range in Figure 5. Clearly, adapted strategies accelerated the learning rate.

By contrast, for interactive participants that did not significantly adapt their policy, the most rapid improvement came around game 21 (mu=20.6, stderr=7.4) meaning it occurred with 95% confidence somewhere in games [6.1, 35.1]. This

range is somewhere during the initial 15 examples and first half of the subsequent self-learning. This aligns with typical learning curves in RL literature where learning occurs at its fastest rate relatively early on, before slowing down and gradually leveling off.

This difference in the location of most rapid learning, between those who did and did not change strategies, is statistically significant: df=5.19, t=4.91, p=0.002.

A second way we can tell that adapted strategies resulted in faster learning is by examining when the performance difference between interactive and batch groups first becomes statistically significant (see Fig. 3). This occurred at game 63 (df=14.1, t=1.79, p=0.048), which again corresponds to when participants' adapted strategies start making up a significant portion of the teaching demonstrations.

B. Interactivity creates a better teaching experience

Interactivity creates a better teaching experience for human players. In particular, it makes players feel more engaged and may encourage them to participate longer (which is inherently beneficial to the machine learner).

1) Longer participation: We asked participants how much longer they would perform the teaching task before they grew bored or tired of it. 70 percent of batch participants reported they tired of the task within the study length, while 30 percent reported interest in continuing longer. Interactive participants, on the other hand, reported qualitatively different results. Many gave length estimates dependent on learner performance.

When batch participants describe how long they would perform the teaching task, all measured length by units of time (*e.g.,* minutes) or by number of games. In other words, when asked how long they would perform the task before tiring of it, they reported things like "a little more, maybe 5 to 10 minutes" or "20 more games".

By contrast, half of the interactive participants (5 of 10) reported variable lengths. Specifically, four reported they would remain interested until the agent performed satisfactorily, and the fifth reported interest until the agent stopped improving. Of the remaining half, three reported a length less than or equal to the study length, and the remaining two reported a length greater than the study length.

If we assume that it takes more than 30 games (the study length) to obtain good performance and that the learner can show steady improvement, then 70 percent of the participants would perform the task longer than the study length. In our hypothetical scenario, this would lead to interactive participants being interested in spending statistically significantly more time on the task (df=1, $\chi^2 = 7.62$, p=0.0058). Unfortunately, our task was not sufficiently challenging. It does however, lead us to speculate that given a sufficiently challenging problem, and a learner that shows steady improvement, future experiments would be able to demonstrate interactivity to encourage longer participation.

2) More engagement: Participants in the interactive group felt more engaged than those in the batch group. We can already see this from previous results on participation length.

In the interactive group, half of the participants described the length of time they would be willing to spend in terms of agent performance, meaning they took stock of the agent and its performance. By contrast, none of the batch participants did. This statistically significant difference (df=1, Chi-squared=10, p-value (two-tailed): 0.0016) suggests a higher level of engagement.

Responses to two other exit interview questions lead us to conclude the interactive group were more engaged. We asked participants whether they "felt like [they] made a difference" and whether they "felt like an active participant". In the interactive group all participants responded positively to both questions. By contrast, in the batch group, 50 percent and 60 percent of participants, respectively, answered positively to the questions. This difference is statistically significant for the difference question (df=1, $\chi^2 = 10$, p=0.0016) and also for the active participant question (df=1, $\chi^2 = 6.7$, p=0.0098).

V. CONCLUSION

We explored the the impact of interactivity on learning performance and teaching experience in a LfD context. In our study where human subjects are asked to teach a computer to play Pac-Man, we found that interactivity improves both learning performance and player experience. In particular, we find interactivity improves the ability of the player to evaluate learner performance, that players changed their teaching strategies based on learner performance, and that resulting adapted strategies resulted in faster learning. With respect to player experience, we find that interactivity makes players feel more like they "made a difference" and more like an active participant. This result is interesting because it implies that the learner is able to improve its own learning environment through transparancy.

Our results show the benefit of tightly coupled interaction to socially guided learning and the importance of SGML considerations when deploying such a system. The results also provide concrete data to help guide practitioners on how best to obtain demonstrations from non-expert humans for LfD applications. It highlights the importance of accurate learner assessments and understanding of the learner to human teachers. We also learn that seeing learner improvement increases teacher engagement and that teacher engagement and interest wanes as learner performance becomes more satisfactory. This suggests that LfD is more suited for obtaining satisfising performance as opposed to refining towards an optimal policy. It also suggests that learning algorithms that can show steady, consistent improvements are more suitable for use with LfD when demonstrations are obtained from non-expert humans.

The first empirical result is particularly interesting because it implies that the learner is able to improve its own learning environment merely through transparancy, without any direct communication channel to the teacher.

REFERENCES

[1] A. Thomaz and C. Breazeal, "Teachable robots: Understanding human teaching behavior to build more effective robot learners," *Artificial Intelligence*, vol. 172, no. 6-7, pp. 716–737, 2008.

[2] L. Vygotskiui, M. Cole, and V. John-Steiner, *Mind in society: The development of higher psychological processes*. Harvard Univ Pr, 1978.

[3] B. Rogoff and J. Lave, *Everyday cognition: Its development in social context*. Harvard University Press Cambridge, MA, 1984.

[4] A. AS, "Nonverbal behavior and nonverbal communication: What do conversational hand gestures tell us?" *Advances in experimental social psychology*, p. 389, 1996.

[5] M. Argyle, R. Ingham, F. Alkema, and M. McCallin, "The different functions of gaze," *Semiotica*, vol. 7, no. 1, pp. 19–32, 1973.

[6] B. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[7] B. Browning, L. Xu, and M. Veloso, "Skill acquisition and use for a dynamically-balancing soccer robot," in *Proceedings of the National Conference on Artificial Intelligence*. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004, pp. 599–604.

[8] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *ICML '04: Proceedings of the twenty-first international conference on Machine learning*. New York, NY, USA: ACM, 2004, p. 1.

[9] B. Price and C. Boutilier, "Accelerating reinforcement learning through implicit imitation," *Journal of Artificial Intelligence Research*, vol. 19, pp. 569–629, 2003.

[10] A. Ng, H. Kim, M. Jordan, S. Sastry, and S. Ballianda, "Autonomous helicopter flight via reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 16, 2004.

[11] R. Aler, O. Garcia, and J. Valls, "Correcting and improving imitation models of humans for robosoccer agents," in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, vol. 3, 2005.

[12] D. Grollman and O. Jenkins, "Dogged learning for robots," in *IEEE International Conference on Robotics and Automation*. Citeseer, 2007, pp. 2483–2488.

[13] S. Chernova and M. Veloso, "Interactive policy learning through confidence-based autonomy," *Journal of Artificial Intelligence Research*, vol. 34, no. 1, pp. 1–25, 2009.

[14] J. Clouse, "On integrating apprentice learning and reinforcement learning," *Electronic Doctoral Dissertations for UMass Amherst*, 1996.

[15] "Pac-man," http://en.wikipedia.org/wiki/Pac-man.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[17] C. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[18] A. Thomaz and C. Breazeal, "Transparency and socially guided machine learning," in *5th Intl. Conf. on Development and Learning (ICDL)*, 2006.

[19] C. Isbell, C. Shelton, M. Kearns, S. Singh, and P. Stone, "A social reinforcement learning agent," in *Proceedings of the fifth international conference on Autonomous agents*. ACM New York, NY, USA, 2001, pp. 377–384.

[20] W. Smart and L. Kaelbling, "Effective reinforcement learning for mobile robots," in *Proceedings- IEEE International Conference on Robotics and Automation*, vol. 4. Citeseer, 2002, pp. 3404–3410.

[21] M. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Proceedings of the eleventh international conference on machine learning*, vol. 157. Citeseer, 1994, p. 163.

[22] R. Crites and A. Barto, "Improving elevator performance using reinforcement learning," *Advances in Neural Information Processing Systems 8*, 1996.

[23] S. Schaal and C. Atkeson, "Robot juggling: An implementation of memory-based learning," *Control Systems Magazine*, vol. 14, no. 1, pp. 57–71, 1994.

[24] P. Cichosz, "An analysis of experience replay in temporal difference learning," *Cybernetics and Systems*, vol. 30, no. 5, pp. 341–363, 1999.

[25] D. Scott, "Parametric statistical modeling by minimum integrated square error," *Technometrics*, vol. 43, no. 3, pp. 274–285, 2001.