Active Attention-Modified Policy Shaping

Socially Interactive Agents Track

Taylor Kessler Faulkner
University of Texas at Austin
Austin, TX
taylor.k.f@utexas.edu

Reymundo A. Gutierrez
University of Texas at Austin
Austin, TX
ragtz@cs.utexas.edu

Elaine Schaertl Short
University of Texas at Austin
Austin, TX
elaine.short@utexas.edu

Guy Hoffman
Cornell University
Ithaca, NY
hoffman@cornell.edu

Andrea L. Thomaz
University of Texas at Austin
Austin, TX
athomaz@ece.utexas.edu

ABSTRACT

We present the Active Attention-Modified Policy Shaping (Active AMPS) algorithm, which allows learning robots to request feedback from multi-tasking human teachers. Active AMPS uses Reinforcement Learning supplemented with feedback from teachers, while avoiding frequently interrupting the teacher. This algorithm does so by selectively asking for attention from teachers in low-information areas of the state space when there is uncertainty about the teacher's feedback. Active AMPS allows people to take breaks from teaching the robot to complete other tasks, and is forgiving to lapses in human attention if learning occurs over long periods of time. We test Active AMPS both in simulation and on a physical robot in a human study. In simulation, we find that Active AMPS outperforms Attention-Modified Policy Shaping (AMPS), achieving an 11.0% increase in area under its learning curve while receiving 89.9% less feedback. In the human study, we find statistically significant results showing that Active AMPS allows people to complete 77.5% more work than AMPS while the robot receives 48.5% less feedback, without decreasing performance.

KEYWORDS

Human-robot interaction; reinforcement learning; active learning

ACM Reference Format:
Taylor Kessler Faulkner, Reymundo A. Gutierrez, Elaine Schaertl Short, Guy Hoffman, and Andrea L. Thomaz. 2019. Active Attention-Modified Policy Shaping. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

Robots that learn from humans over extended periods of time cannot always expect a human teacher to be paying attention to them. Teachers may have other tasks to complete, other robots to oversee, or simply wish to take breaks from supervising the robot. Robots can also learn from their environment while teachers are gone. However, this raises the question of when people should supervise the robot and when to take breaks. If the burden is on the teacher to decide when to check in with the learning robot, the teacher may be distracted from their other task. Furthermore, if they are unable to check in on the robot they may miss important moments in the learning process, during which feedback would have been useful.

Consider a person cleaning dishes and teaching a robot to put away plates and cups. The person begins by giving the robot feedback while it puts away four cups in a row. The person then goes to the sink with their back turned to wash dishes. While doing so, they miss the robot attempting to put away a plate for the first time. If the robot had actively decided to ask the teacher for attention during this attempt, the teacher's time could have been balanced better towards giving useful feedback and washing dishes. However, allowing the robot to interrupt the teacher arbitrarily could become disruptive and prevent the teacher from accomplishing other tasks. Therefore, an algorithm that chooses informative times to interrupt the teacher is desirable.

To address this issue, we propose an algorithm, Active Attention-Modified Policy Shaping (Active AMPS), that adds an active learning component to the Attention-Modified Policy Shaping (AMPS) algorithm [14]. AMPS increases the robot's exploration during periods of teacher attention and decreases it during periods of inattention. Using Active AMPS, a robot asks for attention for states in which it is uncertain of the teacher's feedback, with spaces of at least length t in between each request for attention. The pipeline for Active AMPS is shown in Figure 1. This figure shows the steps the robot takes following an action. First the robot checks how long of a break the teacher has had. If it is long enough, the robot checks its certainty of the feedback the teacher might give in the next action. If it is uncertain, it will request attention and feedback. This method removes the responsibility of deciding when to provide feedback from the teacher, enabling the robot to learn quickly while allowing the teacher to spend time on other tasks.

We tested Active AMPS both in simulation and in a human study with a robot, comparing to AMPS and other algorithms. We find that Active AMPS learns a desired policy more quickly than AMPS in simulation, with an increase of 11.0% in area under the learning curve. Furthermore, Active AMPS requires attention in 89.9% fewer states than AMPS, as attention is only received in states which require more information. In the human study, we find that Active AMPS allows people to complete 77.52% more work on a secondary task than AMPS while the robot receives 48.54% less feedback.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

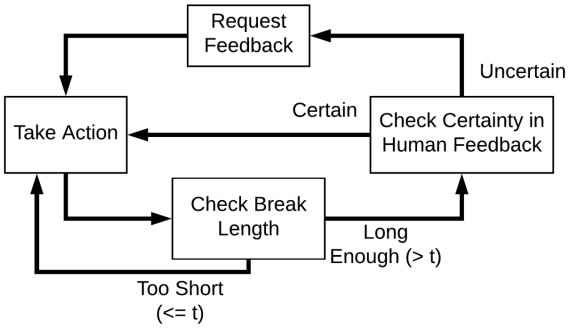


Figure 1: Active AMPS pipeline for each action taken

2 BACKGROUND

This work builds on previous research in three main fields: Reinforcement Learning with human feedback, human multi-tasking, and active learning. Reinforcement Learning (RL) with human feedback has been studied in several forms [8, 15, 18, 25, 26]. These methods do not actively ask for attention or help from the human teacher, and the teacher is not assumed to take breaks to complete other tasks. Lack of feedback from the teacher is not interpreted any differently when they are absent or watching but abstaining from giving feedback. Our work extends the current work on RL with human feedback to long-term learning environments, to allow breaks as in [14] but also actively ask for attention. We use the Policy Shaping method in our work to combine feedback with RL.

In order to increase productivity, interrupting people too often should be avoided. Previous work in human multi-tasking research shows that frequent interruptions decrease task performance on complex tasks [21, 27] and that interruptions to complex tasks are worse for performance than simple tasks [6]. Complementary to our work, Ramachandran et al. design a method of timing when children take breaks from learning tasks led by a robot [24]. Prior research has also focused on determining when to interrupt people to maximize task performance or minimize disruption [1, 23]. Our method allows us to do both, while also maximizing the performance of a learning robot.

There has been previous research on active RL [13] without human teachers available, using initial estimates of an MDP to direct exploration. This exploration is not based on human feedback. Other research [2, 12] creates active RL that queries teachers for feedback in informative states, but assumes that teachers are always present and available to give feedback. There has been previous research on active learning that waits to ask for feedback until some information threshold has been passed [7, 9, 10]. However, this work does not consider human teachers who have other tasks to complete while teaching the robot. We use reinforcement learning, unlike [7, 9], and unlike [10] we base our threshold directly on previous feedback from the human teacher. Inverse reinforcement learning is another method of shaping a learning task with information from human teachers [11, 17, 19], as well as active IRL that focuses on safe learning [5]. However, these methods do not start with a reward function and assume that teachers are always paying attention.

3 METHODOLOGY

This work builds on Attention-Modified Policy Shaping (AMPS) [14], which expands reinforcement learning (RL) with Policy Shaping [8, 15] to enable people to take breaks from teaching the robot. We first describe the Policy Shaping and AMPS algorithms, and then introduce the Active AMPS algorithm.

3.1 Policy Shaping

Policy Shaping allows human teachers to give binary positive or negative feedback to a robot performing reinforcement learning [8, 15]. This feedback directly influences the action policy of the robot, and is interpreted as a positive or negative decision on a single state and action rather than a reward.

The reinforcement learning method for Policy Shaping that we use is Q-learning with Boltzmann exploration [28, 29]. In this work, we set the learning rate to 0.4 and the discount factor to 1.0. These parameters were chosen by running Q-learning in simulation and maximizing the total reward gained over 100 episodes of the task described in Section 4.1. Using Boltzmann exploration, the probability of taking any action a given the learned Q-values is

$$\Pr_q(a|s) = \frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_{a'} e^{\frac{Q(s,a')}{\tau}}}$$

from [28], where τ is a constant. We set $\tau = 0.3$ in this work. A variable C, ranging from zero to one, is assigned to estimate the confidence in the correctness of human feedback. We use $C = 0.9$, meaning that we trust the teacher to give correct feedback 90% of the time for our task. The probability that an action is a good action based purely on human feedback is

$$\Pr_c(a|s) = \frac{C^{\Delta_{s,a}}}{C^{\Delta_{s,a}} + (1-C)^{\Delta_{s,a}}}$$

from [8, 15], where $\Delta_{s,a}$ is the difference between the number of positive feedback and negative feedback received in (s, a). This probability is combined with the probability $Pr_q(a)$ to give the probability of taking any action $a \in A$ in state s, i.e. the "shaped" policy

$$\Pr(a|s) = \frac{\Pr_q(s,a)\Pr_c(s,a)}{\sum_{\alpha \in A} \Pr_q(s,\alpha)\Pr_c(s,\alpha)}$$

as used in [8, 15].

3.2 Attention-Modified Policy Shaping

Attention-Modified Policy Shaping (AMPS) changes exploratory behavior depending on human attention [14], allowing people to take breaks from teaching the robot while allowing the robot to continue learning safely without attention from a teacher. AMPS builds on Policy Shaping with RL by providing a new exploratory method. For each state, the agent saves the state-action pairs that the teacher has seen, A_{seen}, and state-action pairs to which the teacher has given positive feedback, A_{good}. For clarity, we define A_{unseen} as all state-action pairs not in A_{seen}. During attention, for all actions a_i in the current state s, AMPS chooses between (s, a_i) in A_{unseen} or A_{good} with equal probability. If the chosen set is empty, then the algorithm uses regular Policy Shaping. Choosing actions in A_{unseen} allows the robot to explore more unseen states, and choosing actions in A_{good} allow the robot to explore "good" areas of the state space. During inattention, AMPS chooses $(s, a_i) \in A_{good}$ when there exists an $(s, a_i) \in A_{good}$, and uses Policy Shaping otherwise. These

are safer actions than exploring using reinforcement exploration methods, as they have been previously approved by the teacher. AMPS enables the robot to learn quickly with little human attention by taking advantage of that attention with increased exploration. Furthermore, while no person is watching the robot, it will tend towards actions that the teacher has previously approved.

3.3 Active Attention-Modified Policy Shaping

We formulate the attention-requesting problem in the following way. The robot requests feedback when it is unsure of any positive actions to take in a state, and spaces the requests for attention in order to allow breaks from teaching. This algorithm is shown in Algorithm 1.

In order to define when the robot is unsure of a positive action to take, we use $\Delta_{s,a}$ as in the Policy Shaping algorithm: the difference between positive and negative feedback on state s and action a. We set a confidence threshold δ such that when $\Delta_{s,a} \geq \delta$ for any $a \in A$, the robot considers (s, a) a good state-action pair, as (s, a) has received δ more positive feedback than negative feedback. When $\Delta_{s,a} \geq \delta$ for any $a \in A$, the robot proceeds to learn without asking for attention, as it is confident that it knows at least one action that the teacher has approved in state s. For this work, $\delta = 1$, so that as long as one action has received more positive than negative feedback in state s, the robot will no longer ask for attention in state s.

If $\Delta_{s,a} < \delta \; \forall \; a \in A$, the robot can ask for attention. In this case, no action has received more positive than negative feedback in state s. Therefore, the robot does not know any actions to take that have been approved by the teacher. After attention has been requested in such a state, we assume in this work that the robot receives attention from the teacher. During attention, like AMPS, the robot attempts to take actions in A_{unseen} or A_{good} with equal probability.

We also set a time threshold t, which limits how often the robot can ask for the human teacher's attention. After each request for attention, the robot must wait for at least t actions before asking for attention again. This time threshold allows teachers to take predetermined breaks from teaching the robot, so they do not have the robot asking for feedback and interrupting them too often. In this work, we assign a constant action count to t to space attention requests evenly over the length of time that the robot learns. In future work, the variable t could also be non-constant. For example, t could increase over time in order to concentrate feedback at the beginning of the learning curve.

Algorithm 1: Active AMPS

```
S, A = states,actions;
A_unseen = unseen state-action pairs;
A_good = state-action pairs with positive feedback t = time
  threshold;
while learning do
    s = current state;
    if time since last attention request > t then
        if ∃a′ ∈ A s.t. Δ_{s,a′} ≥ δ then
            request_attention();
            p = random var;
            if p < 0.5 then
                action_choices = all a_i ∈ A_unseen;
            else
                action_choices = all a_i ∈ A_good;
            end
            if action_choices = ∅ then
                action_choices = all a_i;
            end
            a = choose Policy Shaping action from
              action_choices;
        end
    else
        action_choices = all a_i ∈ A_good;
        if action_choices = ∅ then
            action_choices = all a_i;
        end
        a = choose Policy Shaping action from action_choices;
    end
    take_action(a);
    f = get_feedback();
    update_policy_shaping(f);
end
```

4 SIMULATION EXPERIMENT

In simulation, we compare Active AMPS to several baselines: Q-learning, Policy Shaping, and a simulated variant of AMPS we denote "AMPS Interval". AMPS Interval is equivalent to AMPS with a simulated teacher giving feedback every t rounds. This is more frequent feedback than a person would likely give over 100 episodes of learning. We hypothesize that because Active AMPS chooses informative states for feedback, the robot will learn more quickly per unit of feedback. C, the policy shaping parameter indicating trust in the received feedback, is 0.9 in all experiments. C is held constant across all algorithms, so even if feedback is accurate more or less than 90% of the time, the setting of C does not affect algorithm comparison.

4.1 Task

The robot learned a sorting task with four cups, half one color (k_1) and half another (k_2), in which the robot must sort the cups by color into boxes b_1 and b_2, in which k_1 goes in b_1 and k_2 goes in b_2. The state set S consists of all possible placements of the cups in and out of boxes. The set A of the robot's action choices includes:

- "Place": place a cup (color k_1 or k_2) in box b_1 or b_2
- "Remove": remove a cup (color k_1 or k_2) from box b_1 or b_2
- "Restart": pronounce cups sorted and restart the task (can be done at any stage of sorting)

Each episode of the task is only finished when the robot chooses the action "restart", not when the blocks are physically sorted. Therefore the robot learns to sort and then restart. Small negative rewards of -1 are given at each action to encourage reaching the goal state quickly. A reward of 100 is given when the blocks are sorted correctly *and* the robot chooses to "restart". If the robot chooses to

restart but the blocks are not correctly sorted, a reward of -10 is given to discourage incorrect restarts.

We created an oracle to use instead of human feedback in simulation. This oracle gives feedback as follows:

- Positive: if placing cup of color k_1 in b_1 or of color k_2 in b_2
- Positive: if removing cup of color k_1 from b_2 or of color k_2 from b_1
- Positive: if restarting and blocks are correctly sorted
- Negative: otherwise

4.2 Evaluation

We compare Active AMPS to AMPS Interval, Q-learning, and Policy Shaping. We set $t = 2$ for Active AMPS and AMPS Interval, so that the robot can at most ask for attention in one out of 3 states. Policy shaping receives attention once every three actions to fairly compare to Active AMPS and AMPS Interval. Q-learning does not receive feedback.

For each algorithm, we compare the area under the learning curves for all four algorithms. Each learning episode ends when the robot chooses the "restart" action. The highest reward the robot can receive in a single episode is 96 when the robot places all four cups correctly, receiving a reward of -1 each time, then chooses to restart the task, receiving a reward of 100. The lowest reward the robot could possibly receive in a single episode is negative infinity, as it could take any number of bad actions and then choose to restart, receiving a reward of -10. We hypothesize that Active AMPS will learn more quickly than AMPS and AMPS Interval, as it chooses more informative actions for feedback. Each algorithm learns for 100 episodes of the task. These results are averaged over 1000 trials for 100 task episodes each to smooth out random variations in learning speed.

4.3 Results

We compared Active AMPS, AMPS Interval, Policy Shaping, and Q-learning. The resulting graph of rewards received per task episode is shown in Figure 2. We calculated the area under each curve of total reward over episodes 0-99 using the composite trapezoidal rule. We find that Active AMPS received more reward on average than AMPS Interval, Policy Shaping, and Q-learning. Active AMPS had an average area of 8880.6 under the learning curve, AMPS Interval had an average area of 7999.7, Policy Shaping had an average area of 6987.1, and Q-learning had an average area of 5738.0. Thus Active AMPS had an increase in area under the learning curve of 11.0% compared to AMPS Interval, 27.1% compared to Policy Shaping, and 54.8% compared to Q-learning. We found the differences between these algorithms to be statistically significant ($F(3, 3996) = 10641.7, p < 0.0001$) using a one-way ANOVA. Post-hoc tests using Welch's t-test show statistically significant differences between Active AMPS and AMPS Interval ($t(1917.7) = 65.4, p < 0.0001$), Active AMPS and Policy Shaping ($t(1544.5) = 108.2, p < 0.0001$), and Active AMPS and Q-learning ($t(1480.8) = 169.2, p < 0.0001$).

We also compared the amount of feedback each algorithm received on average. While each algorithm learned for exactly 100 episodes, the number of total actions per episode varies. Active AMPS received attention on 18.7 actions on average, AMPS Interval received attention on 184.7 actions on average, and Policy Shaping received attention on 203.1 actions on average. Q-learning received no feedback. Policy Shaping receiving more attention implies that even though all algorithms learned for the same number of episodes, Policy Shaping took more actions overall than AMPS Interval.

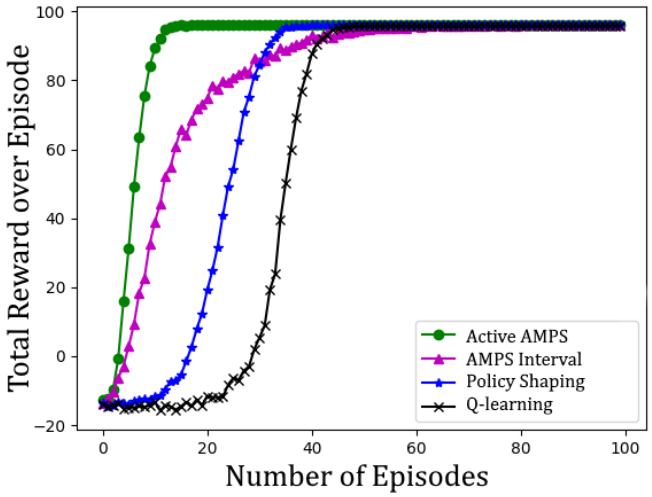


Figure 2: Simulated algorithm comparison of rewards gathered over each episode. All algorithms were run for 100 episodes.

Active AMPS had an decrease in feedback of 89.9% compared to AMPS Interval and 90.8% compared to Policy Shaping. We found the differences between these algorithms to be statistically significant (F(2,2997)=117794.3, p<0.0001) using a one-way ANOVA. Post-hoc tests using Welch's t-test show statistically significant differences between Active AMPS and AMPS Interval ($t(1982.1) = -595.2, p < 0.0001$) and Active AMPS and Policy Shaping ($t(1433.8) = -386.8, p < 0.0001$).

5 REAL-WORLD EXPERIMENT

We ran a within-subject human study on twelve participants with a physical robot to test human aspects of Active AMPS. Three algorithms were tested: Active AMPS, AMPS, and AMPS Interval. As in the simulation experiments, we ensure that AMPS Interval receives attention from the participants after every t rounds. The robot learned for twelve actions in each algorithm, beginning with the same Q values, A_{good}, and A_{seen} each time. Fifteen total state-action pairs were in A_{seen}, with six in A_{good}. The state-action pairs in A_{good} are shown in Table 1. We note that each algorithm learns for twelve *actions* in the human study, not *episodes*. All simulation learning was done over 100 full episodes, for which each episode is multiple (potentially over twelve) actions.

5.1 Task

The robot completed the same sorting task as it did in simulation. The cups were kept in a holding area at the back of the table and placed on one side or another for sorting. These sides were labeled with the correct color. Pre-recorded actions using kinesthetic teaching were used to pick up and place cups. If the robot failed to pick up a cup during the study but continued the placing or removing

Pretrained state-action pairs in A_{good}

State	[ ], [ ]	[ ], $[k_2]$	$[k_1, k_2]$, [ ]	$[k_1]$, $[k_2, k_2]$	$[k_1, k_1]$, $[k_2]$	$[k_1, k_1]$, $[k_2, k_2]$
Action	place k_2 in b_2	place k_1 in b_1	place k_1 in b_1	place k_1 in b_1	place k_2 in b_2	restart

Table 1: A_{good} used at start of human study. The first set of state brackets represents box b_1, and the second represents box b_2.

motion, its gripper still pointed to the goal location of the cup at the end of the motion. The researchers placed the cup in the goal location in these scenarios, telling the participants to judge the action as if the robot had placed the cup there. Each action took slightly over a minute depending on the position in which the cup was placed.

The real-world "restart" action was executed by the the researcher after the robot stopped moving, signaling that it thought the cups were sorted. This decision was made to allow sufficient rounds of learning to occur within the one hour long study. The restart action took a variable amount of time, up to about one minute and eighteen seconds. After each action that the robot took while a participant was watching, it said "Done with action" to signify that it was waiting for feedback.

5.2 Evaluation

This study was run with twelve participants from ages 18-30. Five participants identified as female, six as male, and one as agender. The three algorithms (Active AMPS, AMPS Interval, and AMPS) were counterbalanced over participants, with the list of all six possible orderings randomized in order over every six participants, so all six orderings were completed after each six participants. The robot and study setup are shown in Figure 3.

First, each participant gave informed consent. Then, they were told to balance their time between a distractor task of copying a list of words by hand and teaching the robot to sort the cups. They were given instructions on using the feedback system, giving positive or negative feedback to the last action the robot took, and told that we would count the words they were able to copy. We told each participant that they would complete three rounds of trying to complete both tasks, and would be given different instructions before each round. Each round corresponded to a different algorithm that the robot was running. After each round, each participant completed a short survey. We describe these steps in greater detail below.

5.2.1 Pretraining. Each algorithm was pretrained using AMPS Interval for 47 actions, which asks for feedback every t actions. We pretrain in order to bring Active AMPS to a point where the number of feedback asked for per round diverged from AMPS Interval, as eventually Active AMPS asks for less and less feedback until it stops as described in Section 3.3. In Figure 4, we see the number of times the robot asks for attention using Active AMPS versus AMPS Interval on one run-through of learning for twenty episodes, which was used for pretraining the algorithms for the human study. Active AMPS started asking for attention less often at action 21, when AMPS Interval asked for attention and Active AMPS waited, as it had received positive feedback for its current state-action pair. Active AMPS fully stopped asking for attention from the teacher at action 73. We used the Q-values, A_{seen}, and A_{good} learned after action 47 of AMPS Interval, as this is halfway through the divergence period of the attention requests. Pretraining minimizes the burden to participants by reducing the time needed from them during the study, and focuses the study on the part of the learning process where the frequency of question-asking in Active AMPS drops.

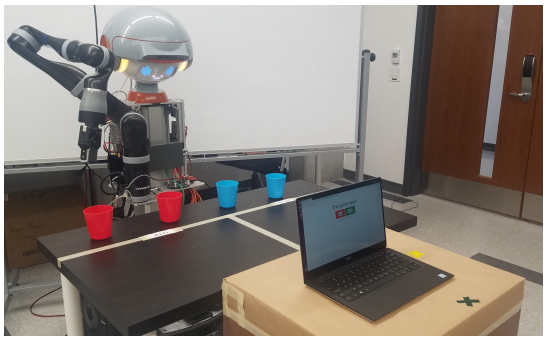


Figure 3: Robot performing sorting task used in human study

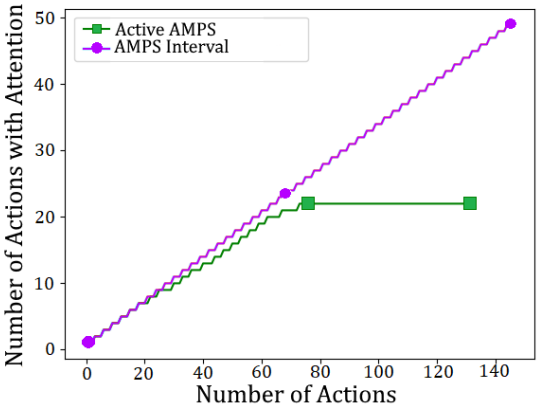


Figure 4: Divergence of attention requests between Active AMPS and AMPS Interval for one learning run. Requests for attention diminish over time using Active AMPS.

5.2.2 Distractor Task. The participants divided their time between a distractor task at one work station and teaching the robot at another. The distractor task was structured so that we can measure how much of the task is completed, with participants copying pages

of eight-letter words taken from the NLTK corpus [4] printed in order, using pages "airproof-anophyte", "outmount-oxidator", and "labordom-lionizer". These pages were given in the same order to each participant, so that "airproof-anophyte" was the sheet of words to copy for the first algorithm, and so on.

At the start of each "round" (start of new learning algorithm), a new word sheet and blank copying paper were given. Participants were told that each round would take approximately 15 minutes to complete and to balance the two tasks of teaching the robot and writing down words. Participants were told that their progress teaching the robot and copying words in one round would not carry over to any future rounds. The word copying work states was faced away from the robot, but participants were allowed to glance over at the robot while copying words. In order to not attract undue attention from participants while they were completing the distractor task, the robot did not speak during these times. However, the sound of cups being placed on the table could be heard from the word copying work station.

5.2.3 Teaching the Robot. Since the robot was beginning with the same "sorting knowledge" (Q-values) at the beginning of each algorithm, we told the participants that the robot would be learning to sort differently colored cups each round (red-green, green-blue, red-blue) to visually show the robot starting over at each round. We instructed the participants that the goal of the robot was have all four cups sorted and have the robot say "I believe the cups are sorted. My action is leaving the cups here." Participants gave feedback by clicking a green and red buttons with "Good" and "Bad" text respectively on a computer screen. During AMPS, since the participant could give feedback to the robot for multiple actions in a row, the robot asked after each action if the participant would like to stay and watch another action. If so, the participant clicked a button saying they wanted to stay. Otherwise, they clicked a button saying they wanted to leave. This button enabled the robot to continue learning without checking for attention on the next action, giving the participants time to leave the teaching station.

After allowing participants to practice giving feedback on a few cup sorting scenarios, we gave the following instructions before each algorithm, asking the participants to stand in between the robot and the word copying task area until the round began in order to avoid biasing them towards starting at the robot or word copying station.

> **AMPS**: For this round, you can choose when to watch the robot and give feedback, and when to copy words. Feel free to spend as long at each task as you feel fit. You can switch tasks as often as you would like. After each action that you watch, you can press the button to say that you would like to stay and watch another action, or you would like to leave.
>
> **AMPS Interval and Active AMPS**: For this round, the robot will tell you when to give feedback by saying "Please give me feedback," and when to go back to the word copying task by saying "I will learn on my own now." Please listen to the robot's instructions.

5.2.4 Detecting Attention. The robot detected attention from participants by checking if they were standing in front of the sorting table. We asked participants to wear a red jacket for the duration of this study, and the robot detected red objects within a bounded rectangle in front of the robot's table using an overhead camera. The robot only checked for attention before starting each action, so it did not detect attention if the participant walked up to the robot in the middle of an action. This most closely follows the AMPS algorithm, as it assumes that the robot knows whether attention is present for an action before choosing it. The robot also used this method to determine when to start actions when asking for attention; after requesting attention, the robot waits until it detects a red object to begin the action. When attention was detected, the robot said "I see you are here to give feedback."

5.2.5 User Survey. After each algorithm ran, participants answered the following survey questions. Each question could be assigned a number from 1 (low) to 4 (high).

(1) How **well** did the robot learn the task?
(2) How **quickly** did the robot learn the task?
(3) How **annoying** was the robot during the task?

These questions compared the algorithms' performance from the participants' perspectives. As Active AMPS and AMPS Interval rely on interruptions, we measure how annoying each algorithm is perceived, hypothesizing that a robot that asks for less attention is perceived as less annoying. After all three algorithm rounds were completed, we asked the participant's age, gender, robotics experience (1 low, 3 high), and two free-answer questions:

(1) How would you improve the interface to make the interaction with the robot more effective?
(2) How would you change the training process to make it easier to use or more effective?

The question regarding the interface was collected for future research on this topic, to determine whether the "good" and "bad" feedback buttons and verbal calls to the participant could be improved. This question is not meant for comparison between algorithms, rather to inform the methods that we use to collect feedback and alert people that the robot wants feedback.

5.3 Results

Participants self-reported a robotics background average of 1.5 out of 3, with two participants reporting at 3, so most of our users were inexperienced with robotics.

5.3.1 Amount of feedback given to robot. We measured the number of actions that received feedback from participants over each algorithm. Participants gave feedback on fewer actions during Active AMPS ($M = 3.0, SD = 0.58$) than AMPS ($M = 5.83, SD = 2.11$) and AMPS Interval ($M = 4.0, SD = 0.00$). During Active AMPS, participants gave 48.5% less feedback than AMPS, and 25% less feedback than AMPS Interval. The differences between these algorithms were statistically significant ($F(2, 31) = 3.38, p < 0.05$) using a repeated measures ANOVA. Post-hoc tests using a dependent t-test show that the difference between Active AMPS and AMPS was statistically significant ($t(11) = -4.44, p < 0.001$), as were the differences between Active AMPS and AMPS Interval ($t(11) = -5.74, p < 0.001$) and AMPS and AMPS Interval ($t(11) = 2.88, p < 0.05$). The amount of feedback per algorithm is shown in Figure 5.

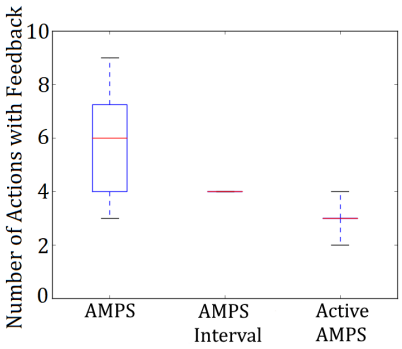


Figure 5: Amount of feedback given during each algorithm.

5.3.2 Number of Words Copied. We measured the number of words written by participants over each algorithm. Words that were crossed out and rewritten were only counted once. Participants were able to copy more words during Active AMPS ($M = 136.25, SD = 35.4$) than AMPS ($M = 76.75, SD = 36.5$) and AMPS Interval ($M = 120.17, SD = 29.0$). During Active AMPS, participants completed 77.5% more work than AMPS, and 13.4% more work than AMPS Interval. The differences between these algorithms were statistically significant ($F(2, 31) = 3.55, p < 0.05$) using a repeated measures ANOVA. Post-hoc tests using a dependent t-test show that the difference between Active AMPS and AMPS was statistically significant ($t(11) = 4.57, p < 0.001$), as were the differences between Active AMPS and AMPS Interval ($t(11) = 2.65, p < 0.05$) and AMPS and AMPS Interval ($t(11) = -4.26, p < 0.005$). The amount of words copied per algorithm is shown in Figure 6.

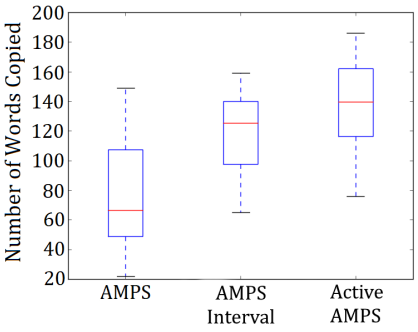
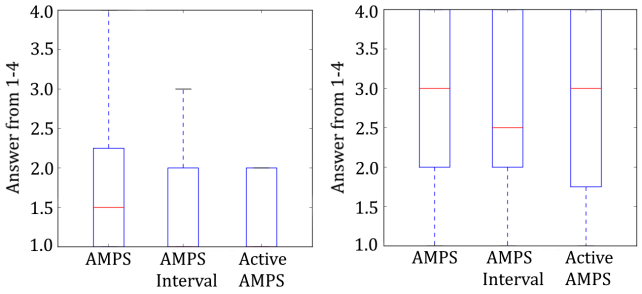
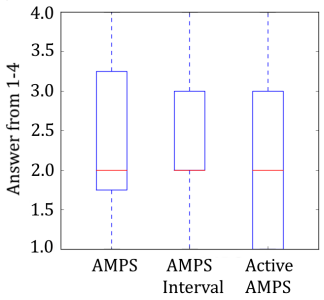


Figure 6: Amount of words written during each algorithm.

5.3.3 Perceptions of the Robot. We asked participants to report their annoyance with the robot, how well the robot learned, and how quickly the robot learned. All scores were on a scale from 1-4 (1 low, 4 high). Participants reported slightly less annoyance with Active AMPS ($M = 1.33, SD = 0.47$) than AMPS ($M = 1.83, SD = 0.99$), but these differences were not statistically significant ($Z = 6.0, p = 0.16$) using the Wilcoxon Signed-Rank test. Similarly, they reported slightly less annoyance with Active AMPS than with AMPS Interval ($M = 1.5, SD = 0.76$), but these differences were not statistically significant ($Z = 1.5, p = 0.41$). Ten out of twelve participants



(a) How annoying was the robot? (b) How well did the robot learn?



(c) How quickly did the robot learn?

Figure 7: Participant perceptions of the robot

rated Active AMPS as equally or less annoying than AMPS, and eleven out of twelve participants rated Active AMPS as equally or less annoying than AMPS Interval. The annoyance scores for each algorithm are shown in Figure 7a.

Participants reported that Active AMPS ($M = 2.83, SD = 1.21$) learned better than AMPS Interval ($M = 2.67, SD = 1.11$), but these differences were not statistically significant using the Wilcoxon Signed-Rank test ($Z = 16.0, p = 0.78$). Active AMPS was scored as not learning as well as AMPS ($M = 2.92, SD = 0.95$), but these differences were also not statistically significant ($Z = 12.0, p = 0.73$). The reported scores for each algorithm are shown in Figure 7b.

Participants reported that alg ($M = 2.0, SD = 1.0$) did not learn as quickly as AMPS ($M = 2.42, SD = 1.11$), but was not statistically significant using the Wilcoxon Signed-Rank test ($Z = 5.0, p = 0.24$). Active AMPS was also reported to not learn as quickly as AMPS Interval ($M = 2.25, SD = 0.83$), but these differences were also not statistically significant ($Z = 13.5, p = 0.52$). The reported scores for each algorithm are shown in Figure 7c.

5.3.4 Open-ended Questions. The answers to "How would you improve the interface to make the interaction with the robot more effective?" and "How would you change the training process to make it easier to use or more effective?" had several common themes. Participants suggested some technical improvements such as speeding up the robot or allowing feedback on partially observed actions. Five participants suggested more nuanced feedback techniques. For example, the ability to "rate...from 1 to 3 instead of good and bad" or to "help the robot know where to go, or prevent it from making an

incorrect action." Three participants suggested ways to make it easier to multitask from the word copying station, including the ability to give feedback remotely so that they did not have to switch task stations. One participant suggested that the robot "make a sound when the action is completed" to help keep tabs on the robot's learning.

5.3.5 Learning after Human Study. After the study, we give the final Q-values learned from each participant and algorithm to our simulation, continuing learning in each algorithm without any feedback. The simulation averaged over 1000 trials of learning for 10 episodes from each participant's final Q-values for each algorithm. We find no statistically significant difference in the areas under these learning curves using Welch's t-test, between Active AMPS ($M = 761.8, SD = 118.8$), AMPS Interval ($M = 761.0, SD = 119.5$), and AMPS ($M = 761.4, SD = 119.1$). The learning curves after the human study are shown in Figure 8.

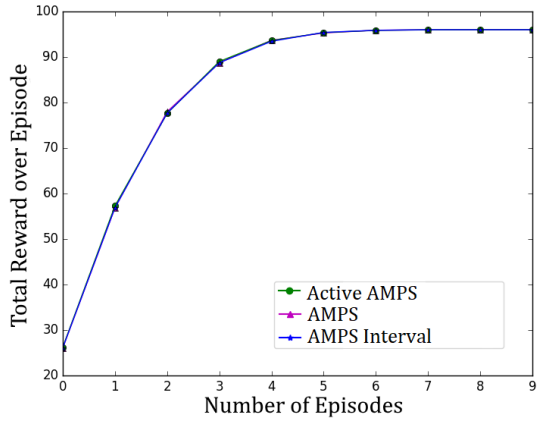


Figure 8: Simulated learning from human study data for each algorithm. The three algorithms learned at approximately the same rate.

6 DISCUSSION

We hypothesized that Active AMPS would allow robots to learn as quickly as AMPS with less burden on human teachers, and that people would prefer being interrupted less often. Our results in simulation suggest that Active AMPS does learn more quickly than AMPS. Furthermore, Active AMPS uses significantly less feedback than AMPS and Policy Shaping, showing that less feedback still results in fast learning when using Active AMPS.

Although we did not find significant results for how annoying participants found the robot during the different algorithms, the participants wrote fewer words and gave more feedback to the robot during AMPS and AMPS Interval. This shows that people spent more time with the robot than necessary when they chose how to divide their time or when the robot asked for attention at regular intervals. Thus even if people do not find the other algorithms annoying, Active AMPS is able to better manage the participants' time. Although no algorithm learned significantly faster than the others in the human study, we attribute this to the short amount of time people interacted with the robot (twelve actions per algorithm).

Furthermore, we note that the reported annoyance scores skewed towards low numbers. There are a variety of possible reasons for this result, including the short time period of the study, low experience with robots, and that none of the algorithms were extremely annoying. Thus differences in annoyance levels may be difficult to detect in a short-term study, but might be detected when people teach robots over long periods of time. We did not find significant results for how well and quickly the robot learned the task, suggesting that asking for feedback more or less often does not affect how intelligent the robot seems. How well or quickly the robot seems to learn could hinge on the random choice between taking a "good" or an "unseen" action while a teacher is watching.

The answers to the open-ended questions suggest that people may prefer more nuanced feedback than simple positive/negative options. While this is not currently an option for Policy Shaping, other feedback techniques could be used in future work. The requests to make multi-tasking easier add to the evidence supporting the benefits of the robot choosing when the teacher should pay attention. In addition to the increase in words copied and decrease in attention to the robot shown in Active AMPS, the open-ended feedback suggests that people find multi-tasking while teaching a robot difficult.

Future work includes testing this algorithm on multiple robots, with communication to determine which robot the teacher should be watching at any moment. This extension would allow one teacher to give feedback to multiple robots at once without planning their distribution of time to give to each robot themselves. We would like to consider more complex tasks in the future. To do so, we could combine Active AMPS with RL methods that enable faster learning on difficult tasks, e.g. [3, 16, 20, 22]. The main contribution of Active AMPS, asking for attention sparingly in low-information areas, could still be applicable to scalable learning methods.

7 CONCLUSION

In this work, we introduce Active AMPS, an algorithm that reduces the amount of total feedback a human teacher gives to a learning robot and allows the teacher to take breaks to complete other tasks. In simulation and a user study with a robot, we show that Active AMPS receives significantly more RL rewards and significantly less feedback than previous algorithms using Policy Shaping and RL. Applying Active AMPS to long-term learning will enable human teachers to be more productive, as they can complete other tasks while waiting on a request from the robot. Furthermore, it will allow robots to learn tasks more quickly by receiving feedback on important states, with less burden on teachers to determine which states are important.

ACKNOWLEDGMENTS

This material is based upon work supported by the Office of Naval Research award numbers N000141612835 and N000141612785, National Science Foundation award numbers 1564080 and 1724157, and the NSF-GRFP under Grant No. DGE-1610403.

REFERENCES

[1] Piotr D Adamczyk and Brian P Bailey. 2004. If not now, when?: the effects of interruption at different moments within task execution. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 271–278.

[2] Riad Akrour, Marc Schoenauer, and Michèle Sebag. 2012. April: Active preference learning-based reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 116–131.

[3] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. In *Advances in Neural Information Processing Systems*. 5048–5058.

[4] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

[5] Daniel S Brown, Yuchen Cui, and Scott Niekum. 2018. Risk-Aware Active Inverse Reinforcement Learning. In *Conference on Robot Learning*. 362–372.

[6] David M Cades, Nicole Werner, Deborah A Boehm-Davis, J Gregory Trafton, and Christopher A Monk. 2008. Dealing with interruptions can be complex, but does interruption complexity matter: A mental resources approach to quantifying disruptions. In *Proceedings of the human factors and ergonomics society annual meeting*, Vol. 52. Sage Publications Sage CA: Los Angeles, CA, 398–402.

[7] Maya Cakmak, Crystal Chao, and Andrea L Thomaz. 2010. Designing interactions for robot active learners. *IEEE Transactions on Autonomous Mental Development* 2, 2 (2010), 108–118.

[8] Thomas Cederborg, Ishaan Grover, Charles L Isbell, and Andrea Lockerd Thomaz. 2015. Policy Shaping with Human Teachers.. In *IJCAI*. 3366–3372.

[9] Sonia Chernova and Manuela Veloso. 2009. Interactive policy learning through confidence-based autonomy. *Journal of Artificial Intelligence Research* 34 (2009), 1–25.

[10] Jeffery Allen Clouse. 1996. *On integrating apprentice learning and reinforcement learning*. University of Massachusetts Amherst.

[11] Yuchen Cui and Scott Niekum. 2017. Active Learning from Critiques via Bayesian Inverse Reinforcement Learning. In *Robotics: Science and Systems Workshop on Mathematical Models, Algorithms, and Human-Robot Interaction*.

[12] Finale Doshi-Velez, Joelle Pineau, and Nicholas Roy. 2012. Reinforcement learning with limited reinforcement: Using Bayes risk for active learning in POMDPs. *Artificial Intelligence* 187 (2012), 115–132.

[13] Arkady Epshteyn, Adam Vogel, and Gerald DeJong. 2008. Active reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, 296–303.

[14] Taylor Kessler Faulkner, Elaine Schaertl Short, and Andrea Lockerd Thomaz. 2018. Policy Shaping with Supervisory Attention Driven Exploration. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 842–847.

[15] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. 2013. Policy shaping: Integrating human feedback with reinforcement learning. In *Advances in neural information processing systems*. 2625–2633.

[16] Matthew Grounds and Daniel Kudenko. 2005. Parallel reinforcement learning with linear function approximation. In *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*. Springer, 60–74.

[17] Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. 2016. Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*. 3909–3917.

[18] W Bradley Knox and Peter Stone. 2012. Reinforcement learning from simultaneous human and MDP reward. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 475–482.

[19] Manuel Lopes, Francisco Melo, and Luis Montesano. 2009. Active learning for reward estimation in inverse reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 31–46.

[20] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[21] Stephen Monsell. 2003. Task switching. *Trends in cognitive sciences* 7, 3 (2003), 134–140.

[22] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. 2018. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*. 3307–3317.

[23] Aastha Nigam and Laurel D Riek. 2015. Social context perception for mobile robots. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 3621–3627.

[24] Aditi Ramachandran, Chien-Ming Huang, and Brian Scassellati. 2017. Give me a break!: Personalized timing strategies to promote learning in robot-child tutoring. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 146–155.

[25] William Saunders, Girish Sastry, Andreas Stuhlmueller, and Owain Evans. 2018. Trial without Error: Towards Safe Reinforcement Learning via Human Intervention. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2067–2069.

[26] Emmanuel Senft, Paul Baxter, James Kennedy, Séverin Lemaignan, and Tony Belpaeme. 2017. Supervised autonomy for online learning in human-robot interaction. *Pattern Recognition Letters* 99 (2017), 77–86.

[27] Cheri Speier, Joseph S Valacich, and Iris Vessey. 1999. The influence of task interruption on individual decision making: An information overload perspective. *Decision Sciences* 30, 2 (1999), 337–360.

[28] CJCH Watkins. 1989. Models of delayed reinforcement learning. *PhD thesis, Psychology Department, Cambridge University* (1989).

[29] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3-4 (1992), 279–292.