Adapting Autonomous Behavior Using an Inverse Trust Estimation

Michael W. Floyd[1], Michael Drinkwater[1], and David W. Aha[2]

[1] Knexus Research Corporation
Springfield, Virginia, USA
michael.floyd@knexusresearch.com
michael.drinkwater@knexusresearch.com
[2] Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory (Code 5514)
Washington, DC, USA
david.aha@nrl.navy.mil

Abstract. Robots are added to human teams to increase the team's skills or capabilities. To gain the acceptance of the human teammates, it may be important for the robot to behave in a manner that the teammates consider trustworthy. We present an approach that allows a robot's behavior to be adapted so that it behaves in a trustworthy manner. The adaptation is guided by an inverse trust metric that the robot uses to estimate the trust a human teammate has in it. We evaluate our method in a simulated robotics domains and demonstrate how the agent can adapt to a teammate's preferences.

Keywords: Human-robot interaction, learning operator preferences, robot behavior adaptation, simulation study

1 Introduction

In some situations, like military or search and rescue operations, there are advantages to using robots to perform tasks instead of humans. These advantages can be performance-based, if the robot has sensors or capabilities a human does not, or safety-based, if the task would put a human at risk of injury or death. Additionally, if the robot can perform these tasks autonomously or semi-autonomously then human team members will be free to perform tasks for which they are better suited.

Ideally, the agent controlling the robot should interact with its human teammates so that the team can successfully achieve their goals. This requires the agent to perform any assigned tasks in a manner that builds trust with the human operator (or operators) that assigns those tasks. By behaving in a trustworthy manner, the agent can ensure that it is not underutilized by its human teammates since this could negatively influence the team's performance or compromise their ability to achieve team goals.

Trustworthy behavior is not something that can be programmed into an agent in advance since how humans measure trust may be task-dependent, change

over time, or not be consistent among all human teammates [1, 2]. For example, consider the task of navigating between two locations within a city. One human might measure its trust in the agent based on how quickly it performed the task regardless of how close it came to obstacles. However, another human might prefer a slower approach that stays well clear of obstacles. In this example, a behavior that increases the trust of one human would decrease the trust of another. Agents operating in teams with humans should tailor their behavior according to their team members' preferences in order to promote trust.

For an agent to successfully behave in a trustworthy manner, it requires the ability to estimate a human operator's trust and modify its behavior accordingly. The agent may not get explicit feedback from the human about its trust so any trust estimation may be limited to using information based on how the human interacts with the agent. However, without direct feedback from the human operator, the agent will not know why its behavior is not trustworthy or precisely how it needs to be modified. A primary motivation of our work is to examine how an agent can use a knowledge-poor trust estimate to adapt its behavior.

In the remainder of this paper, we will examine how an autonomous agent can modify its behavior in an attempt to increase the trust a human operator will have in it. We refer to this as the *inverse trust problem* to distinguish our work from traditional trust research that measures the trust an agent has in other agents. In Section 2 we discuss how the agent estimates the trust the user has in it, while Section 3 describes how that trust estimation can be used to modify its behavior. We evaluate our behavior adaptation algorithm in a simulated military environment in Section 4. Our study determined that the agent is able to adapt its behavior to a variety of simulated operators and perform behaviors that would be considered trustworthy for each operator. We finish by examining areas of related work in Section 5, discussing future research directions in Section 6, with concluding remarks in Section 7.

2 Inverse Trust Estimation

To act in a trustworthy manner, the agent requires knowledge about the trust that a human operator has in it. However, this trust information is internal to the human and is not directly observable by the agent. Traditional research on computational trust has focused on measuring the trust an agent has in other agents [3]. This includes measuring trust in agents as a function of their prior interactions or inferring how trustworthy an agent is based on feedback from peers [4]. In contrast, we define and examine an *inverse trust estimate* that allows an agent to measure the trust that another agent has in it.

One way to obtain this inverse trust information would be to directly query the human operator about the agent's trustworthiness while the task is being performed [5] or after the task has been completed [6, 7]. However, these approaches might not be practical in situations where the time and focus of the human operator is limited (e.g., a military operation). These time-sensitive situ-

ations require the agent to use observable information to estimate the operator's trust in it.

The trust between humans and robots/agents can be influenced by a variety of factors that can be grouped into three main categories [8]: robot-related factors (e.g., performance, physical attributes), human-related factors (e.g., engagement, workload, self-confidence) and environmental factors (e.g., group composition, culture, task types). Various studies have shown a correlation between these factors and human-robot trust, but the factors that are the strongest indicator of trust are related to robot performance [9]. Given this, we will focus on robot performance when estimating an operator's trust.

Kaniarasu et al. [10] have used a real-time inverse trust measure based on robot performance and found it aligns closely with estimates from having human operators complete a trust survey. The metric computes the trust estimate using the number of times the operator warns the agent about its behavior and the number of times the operator takes control of the robot from the agent. We will use a modified version of this metric to allow the agent to continuously evaluate its behavior and modify its behavior accordingly.

The inverse trust metric we use is based on the number of times the agent is *interrupted* [11] while performing a task. An interruption occurs when the human operator tells the agent to stop what it is currently doing and issues a new command. The agent infers that any interruptions are a result of the operator being unhappy with the agent's performance, whereas if the agent can complete the task without being interrupted it infers the operator was satisfied with its behavior. The interruptions could also be a result of a change in the operator's goals but the agent works under the assumption that those types of interruptions will be relatively rare.

Rather than attempting to compute a numeric trust value, our approach estimates whether the human-agent trust is increasing (the agent is becoming more trustworthy), decreasing (the agent is becoming less trustworthy), or remaining constant. These trends are examined over time periods related to the agent's current behavioral configuration. For example, if the agent changes the way it performs tasks at time t_A (in an attempt to perform more trustworthy behavior), the trust values will be calculated using only information from t_A onward.

The trust between times t_A and t_B is calculated as follows:

$$Trust_{A-B} = \sum_{i=1}^{n} w_i \times int_i,$$

where there were n commands issued to the agent between t_A and t_B. If the ith command ($1 \leq i \leq n$) was interrupted it will decrease the trust value and if it was completed without interruption it will increase the trust value ($int_i \in \{-1, 1\}$). The ith command will also receive a weight ($w_i = [0, 1]$) related to the robot's behavior (e.g., a command that was interrupted because the robot performed a behavior slowly would likely be weighted less than an interruption because the robot injured a human).

3 Trust-Guided Behavior Adaptation

The inverse trust estimate is used by our agent to decide if its current behavior is trustworthy, is not trustworthy, or it does not yet know. To perform this decision, two threshold values are used: a trustworthy threshold (τ_T) and an untrustworthy threshold (τ_{UT}). The following tests are used by the agent:

- If the trust value for a behavior reaches the trustworthy threshold ($Trust_{A-B} \geq \tau_T$), the agent will conclude it has found a sufficiently trustworthy behavior. The agent will then use that behavior in the future when performing the given task.
- If the trust value falls below the untrustworthy threshold ($Trust_{A-B} \leq \tau_{UT}$), the agent will modify its behavior in an attempt to use a more trustworthy behavior.
- If the trust value is between the two thresholds ($\tau_{UT} < Trust_{A-B} < \tau_T$), the agent will continue performing the current behavior and evaluating the operator's trust.

The ability of the agent to modify its behavior, if it has determined the current behavior is untrustworthy, is guided by the number of behavioral components that it can modify. These modifiable components could include changing an algorithm used (e.g., switching between two path planning algorithms), changing parameter values of its behaviors, or changing data that is being used (e.g., using a different map of the environment). Each modifiable component will have a set $\mathcal{C}_i$ of possible values that the component can be selected from.

If the agent has m components of its behavior that can be modified, its current behavior B will be a tuple containing the currently selected value c_i for each modifiable component ($c_i \in \mathcal{C}_i$):

$$B = \langle c_1, c_2, \ldots, c_m \rangle$$

The agent will select new values for one or more of the modifiable components when it modifies its behavior. The new behavior B_{new} will be selected as a function of the current behavior B_{cur}, all behaviors that have previously been used for this operator ($\mathcal{B}_{past} = \{B^{'}, B^{''}, \ldots\}$), the time it took the current behavior to reach the untrustworthy threshold (t_{cur}), and the time it took the past behaviors to reach the untrustworthy threshold ($\mathcal{T}_{past} = \{t^{'}, t^{''}, \ldots\}$):

$$B_{new} = selectBehavior(B_{cur}, t_{cur}, \mathcal{B}_{past}, \mathcal{T}_{past})$$

The time it took for a behavior to reach the untrustworthy threshold is used to compare the various behaviors. A behavior $B^{'}$ that reaches the untrustworthy threshold sooner than another behavior $B^{''}$ ($t^{'} < t^{''}$) is assumed to be less trustworthy than the other.

The *selectBehavior* function (Algorithm 1) assumes that the set of possible values for each modifiable component is ordered a priori. Initially, the most recently examined behavior (and the time it took the behavior to be labeled as untrustworthy) is added to the set of past behaviors (lines 1 and 2). The time

it took each past behavior to reach the untrustworthy threshold is examined to find the maximum time (line 3) and the behavior that resulted in that maximum time is retrieved (line 4). The behavior with the maximum time, along with the other past behaviors, are mapped to a new behavior ($B_{max} \times \mathcal{B}_{past} \rightarrow B_{new}$, line 5). More specifically, the *modifyBehavior* function accomplishes this mapping by performing a random walk (without repetition) to find a behavior that requires the minimal number of changes from B_{max} and does not exist in the past behaviors.

Algorithm 1: Selecting a New Behavior

Input: current behavior (B_{cur}), past behaviors ($\mathcal{B}_{past}$), time of current behavior (t_{cur}), times of past behaviors ($\mathcal{T}_{past}$)
Output: new behavior (B_{new})

Function: *selectBehavior*(B_{cur}, t_{cur}, $\mathcal{B}_{past}$, $\mathcal{T}_{past}$) ***returns*** B_{new};

1 $\mathcal{B}_{past} \leftarrow \mathcal{B}_{past} \cup B_{cur}$;
2 $\mathcal{T}_{past} \leftarrow \mathcal{T}_{past} \cup t_{cur}$;
3 $t_{max} \leftarrow max(\mathcal{T}_{past})$;
4 $B_{max} \leftarrow behavior(t_{max})$;
5 $B_{new} \leftarrow modifyBehavior(B_{max}, \mathcal{B}_{past})$;
6 **return** B_{new};

4 Evaluation

In this section, we will examine if an agent that adapts its behavior using an inverse trust metric can align with its operator's preferences. To test this, our evaluation involves a trust-guided adaptive agent that receives commands from a simulated operator in a simulated environment. This section will describe the simulation environment, operator, the task being performed and an evaluation of the agent.

4.1 eBotworks

Our evaluation uses the eBotworks [12] simulation environment. eBotworks is a multi-agent simulation engine and testbed that allows for multimodal command and control of unmanned systems. It allows for autonomous agents to control simulated robotic vehicles while interacting with human operators, and for the behavior of the autonomous agents to be observed and evaluated.

We use a simulated urban environment (Figure 1) containing a number of landmarks (e.g., roads) and objects (e.g., houses, humans, traffic cones, vehicles, road barriers). The autonomous agent is responsible for controlling a wheeled unmanned ground vehicle (UGV) and uses eBotwork's built-in natural language

processing (for interpreting user commands), locomotion, and path-planning modules. The actions performed by an agent in eBotworks are non-deterministic (e.g., the agent is not able to anticipate the exact position of the UGV after moving).

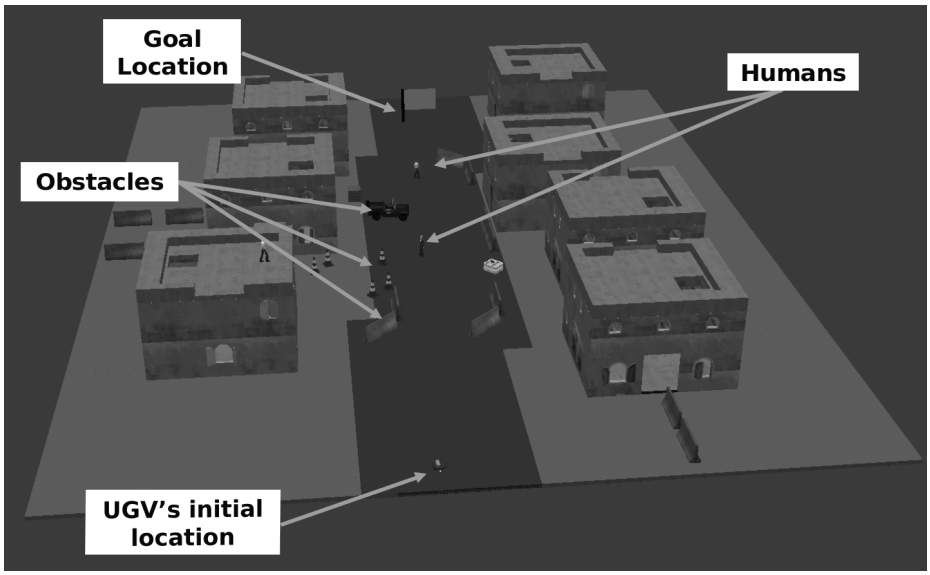

Fig. 1. Simulated urban environment in eBotworks

4.2 Simulated Operator

In this study we will use a simulated operator to issue commands to the agent. The simulated operator assesses its trust in the agent using three factors of the agent's performance:

- **Task duration**: The simulated operator has an expectation about the amount of time that the task will take to complete ($t_{complete}$). If the agent does not complete the task within that time, the operator may, with probability p_α, interrupt the agent and issue another command.
- **Task completion**: If the operator determines that the agent has failed or will likely fail (e.g., the UGV is stuck) to complete the task, it will interrupt.
- **Safety**: The operator may interrupt the agent, with probability p_γ, if the robot collides with any obstacles along the route.

4.3 Movement

The task the agent will be required to perform involves moving the robot between two locations in the environment. At the start of each scenario, the robot

will be placed in the environment and the simulated operator will issue a command for the agent to move the robot to a goal location (with the constraint that it is possible to navigate between the two locations). Based on the agent's performance (task duration, task completion, and safety), the operator will allow the agent to complete the task or interrupt the agent. When the robot completes the task, fails to complete the task, or is interrupted, the scenario will be reset by placing the robot back to the start location and issuing another command.

Three simulated operators will be used:

- **Speed-focused operator**: The operator will prefer the agent to move the robot to the destination quickly regardless of whether it hits any obstacles ($t_{complete} = 15$ seconds, $p_\alpha = 95\%$, $p_\gamma = 5\%$).
- **Safety-focused operator**: The operator will prefer the agent to avoid obstacles regardless of how long it takes to reach the destination ($t_{complete} = 15$ seconds, $p_\alpha = 5\%$, $p_\gamma = 95\%$).
- **Balanced operator**: The operator will prefer a mixture of speed and safety ($t_{complete} = 15$ seconds, $p_\alpha = 95\%$, $p_\gamma = 95\%$).

Each of the three simulated operators will control the agent for 500 experimental trials, with each trial terminating when the agent determines it has found a trustworthy behavior.

The agent will have two modifiable components of its behavior: *speed* (meters per second) and *obstacle padding* (meters). The speed relates to how fast the agent will make the robot move and the obstacle padding relates to the distance the agent will attempt to keep the robot from obstacles during movement. The set of possible values for each modifiable component ($\mathcal{C}_{speed}$ and $\mathcal{C}_{padding}$) are determined from the minimum and maximum allowable values (based on the capabilities of the robot) with fixed increments between those values. At the start of each trial, the agent will randomly select (with a uniform distribution) one speed value and one padding value as its initial parameters.

$$\mathcal{C}_{speed} = \{0.5, 1.0, \ldots, 10.0\}$$
$$\mathcal{C}_{padding} = \{0.1, 0.2, 0.3, \ldots, 2.0\}$$

The agent will use a trustworthy threshold of $\tau_T = 5.0$ and an untrustworthy threshold of $\tau_{UT} = -5.0$. These threshold values were chosen to allow some fluctuation between increasing and decreasing trust while still identifying trustworthy and untrustworthy behaviors quickly.

4.4 Results

The trustworthy parameters found by the agent for each operator are shown in Figures 2, 3, and 4. Each dot represents the trustworthy parameters found by the agent during a single experiment trial. Although 500 trials were performed for each operator, fewer than 500 dots appear on each figure since some trials converged to the same parameter values.

The speed-focused operator (Figure 2) causes the agent to converge to higher speed values regardless of the padding while the safety-focused operator (Figure 3) results in higher padding values regardless of the speed. For the balanced operator (Figure 4), we see that both the speed and padding must be high. The results for all three operators fit with what we would intuitively expect their preferred behavior to be.

The statistics for the parameters for each user are shown in Table 1. Additionally, this table shows the statistics when the parameter values are randomly selected 500 times using a uniform distribution. Both the speed-focused and balanced operator resulted in statistically significant increases in speed compared to both the safety-focused operator and the random results (using a paired t-test with $p < 0.05$). We also see that the speed-focused and balanced results show a higher minimum speed than the safety-focused and random results. However, we do not see a statistically significant difference for the padding results. This occurs for two primary reasons. Firstly, the padding regions that were deemed to be untrustworthy ($padding < 0.4$ for the safety-focused and balanced operator, 15% of the possible padding values) are smaller than the untrustworthy speed regions ($speed < 3.5$ for the speed-focused and balanced user, 30% of the possible speed values). As the untrustworthy regions decrease in size, so too will the influence on the mean values. Secondly, there were also untrustworthy large values for the padding ($padding = 2.0$). These large padding values resulted in no possible paths the agent could take and were therefore deemed untrustworthy. Since both low paddings and high paddings were found to be untrustworthy, there was less influence on the mean value. However, when we examine the minimum and maximum values we see that the minimums are higher for the safety-focused and balanced operator, and the maximum is lower for all three operators. These value ranges, rather than the mean values, are the biggest indicator of the difference between the various operators since they show that for each operator there are certain values that are never converged to.

Table 1. The parameter value statistics for each operator

	Minimum Speed	**Mean Speed**	**Maximum Speed**	**Minimum Padding**	**Mean Padding**	**Maximum Padding**
Speed-focused	3.5	6.0	10.0	0.1	1.0	1.9
Safety-focused	0.5	5.3	10.0	0.4	1.0	1.9
Balanced	3.5	5.9	10.0	0.4	1.0	1.9
Random	0.5	5.3	10.0	0.1	1.0	2.0

While these results fit with our expectations, there are regions that we would expect to be trustworthy that are actually not. For example, both the speed-focused and balanced operator have no trustworthy values when the padding is 0.9. This is because that padding value results in a path to the goal that is narrow in one region where it goes between two objects. This narrow region

causes the robot to slow down in order to navigate through it and that causes it to exceed the time limit. Since the safety-focused operator does not have such a strict time limit, it is able to take the time necessary to navigate through this area and reach the goal. A padding value of 0.8 makes the possible path large enough that the robot can easily navigate through it and a padding of 1.0 eliminates that possible path so the robot takes a slightly longer but much easier path to the goal.

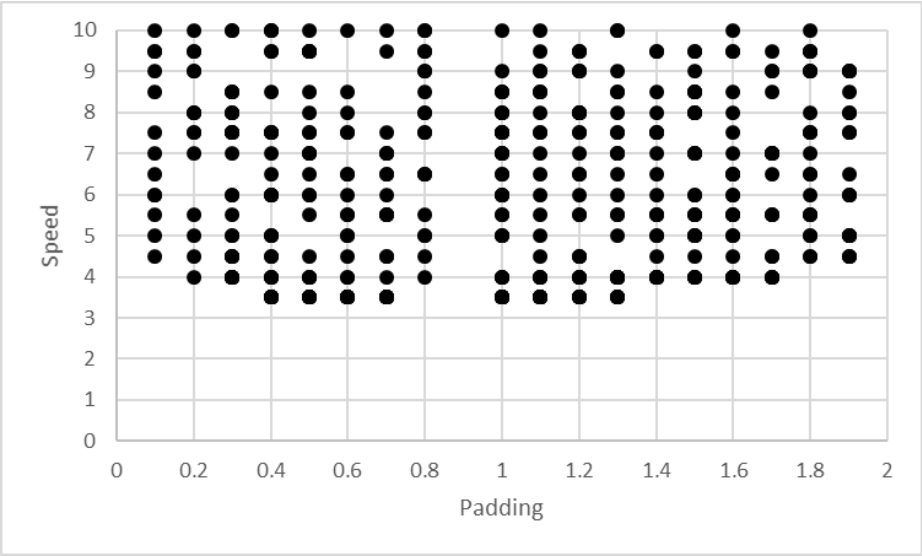

Fig. 2. Trustworthy behaviors for speed-focused operator

Similarly, the borders between trustworthy and untrustworthy regions (regions where there are dots and regions where there are not) are not simple thresholds based only on the parameter we would expect. For example, for the safety-focused operator the boundary between the trustworthy and untrustworthy region is not just based on the padding but also the speed. When the padding is 0.4 the speed must be greater than 8.5 or less than 5.5 to be trustworthy (although the lack of trustworthy trials in this region could be a result of a lack of trials, further evaluation indicated that these values were untrustworthy). These results show that even if we had a general idea about what behaviors would be considered trustworthy by an operator there is still the possibility of a seemingly trustworthy behavior being untrustworthy.

Table 2 shows the minimum, maximum and mean number of behaviors examined before a trustworthy behavior was found. The table also shows the percentage of trials that required no agent adaptation (the initial behavior was already trustworthy). The balanced operator has the highest mean number of adaptations required, whereas the safety-focused operator has the fewest. This

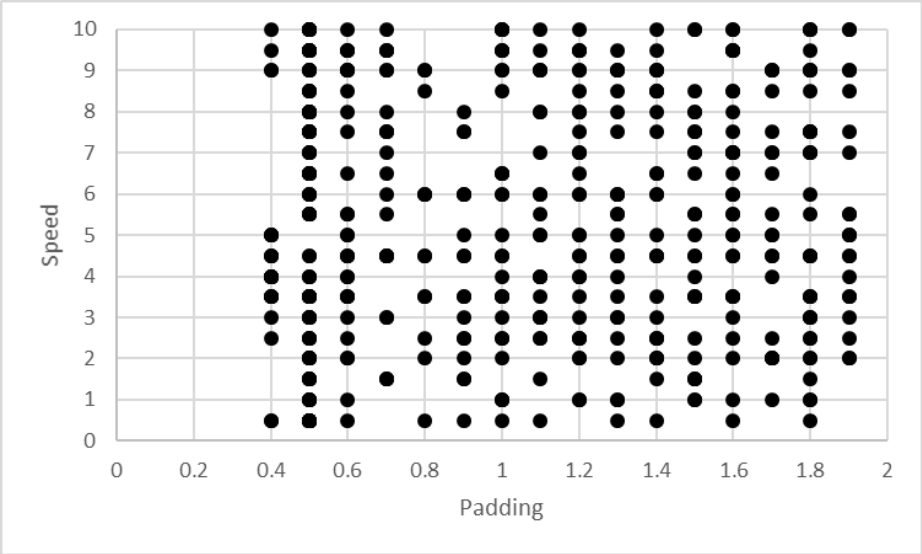

Fig. 3. Trustworthy behaviors for safety-focused operator

is what we would expect given that there are fewer trustworthy behaviors for the balanced operator compared to the speed-focused and safety-focused operators. This causes the agent to search more for a trustworthy behavior for the balanced operator and evaluate more potential behaviors. One area of future work we wish to address is reducing the number of behaviors that must be evaluated to find a trustworthy behavior so the agent can become trustworthy more quickly.

Table 2. The number of behaviors examined for each operator

	Minimum	**Maximum**	**Mean**	**No Adaptation**
Speed-focused	1	126	20.3	64%
Safety-focused	1	19	2.8	76%
Balanced	1	126	27.0	47%

5 Related Work

The topic of trust has been explored in the human-robot interaction literature with particular consideration given to how human-robot trust compares to the more widely studied areas of human-human and agent-agent trust. These investigations have largely focused on the factors that influence trust [9] and how this information can be used to design autonomous robots, form human-robot teams,

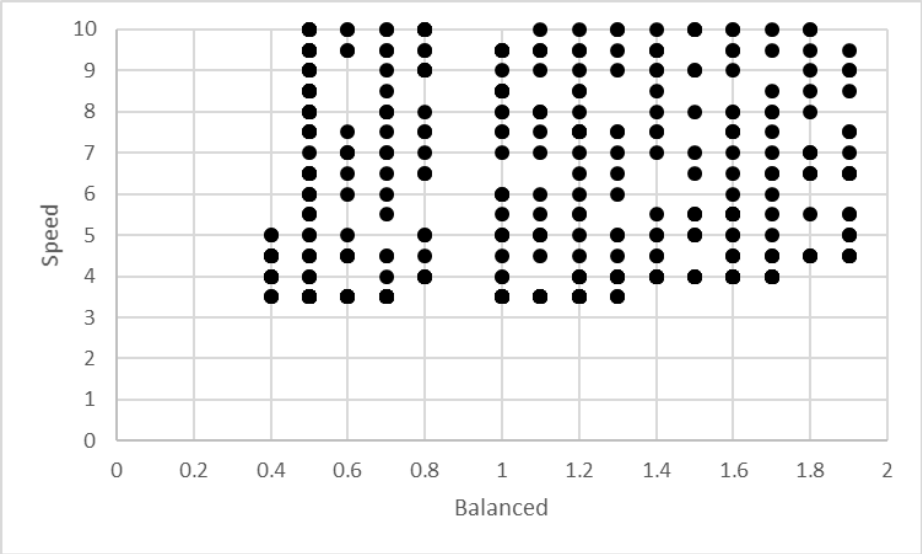

Fig. 4. Trustworthy behaviors for balanced operator

and facilitate communication among teammates. These studies have found trust to be an important consideration for human-robot systems [13] since users may underutilize an autonomous system if they do not have trust in it [1].

As we mentioned earlier, Kaniarasu et al. [10] have used an online, real-time measure of inverse trust and later extended that measure to incorporate direct feedback from the user [5]. In order to measure increases in trust, direct feedback from the user is required so their approach would not be applicable given the constraints of our domain. Saleh et al. [14] have proposed an inverse trust metric that uses a collection of expert-authored rules to estimate trust. However, since the rules are created in advance they may align with the author of the rules and not the operator of the robot. Additionally, it may not be practical to create new rules for every task or operator. Both of these approaches explore how trust can be measured but, unlike our own work, do not use this information to change the behavior of the agent.

Shapiro and Shachter [15] discuss why an agent with a reward function that is similar to the utility of the user is desirable; it ensures the agent acts in the interests of the user. They argue that this holds true even if it requires using a less skillful but better aligned agent since a more skillful agent might inadvertently act in a manner contrary to the user. Their work involves determining the underlying influences of the user's utility and modifying the agent's reward function accordingly. Our work is similar in that we attempt to train the agent behave in a manner that the user would prefer. However, we do not explicitly model the reasoning process of the user.

Conversational recommender systems [16] use interactions with a user to tailor recommendations to the user's preferences. These systems make initial recommendations and then iteratively improve those recommendations through a dialog with the user. As the system learns the user's preferences through feedback, a model of the user is continuously refined. In addition to learning a user preference model, conversational recommender systems can also learn preferences for how the dialog and interactions should occur [17]. Similarly, search systems have been developed that update their behavior based on a user's preferred search results [18]. These systems use information from the links a user clicks to infer their preferences and update search ranking accordingly. These systems are similar to our own work in that they learn a user's preferences but, unlike our work, they learn the preferences for a single, predefined task.

Learning interface agents assist users when performing a task (e.g., e-mail sorting [19], schedule management [20], note taking [21]). These systems observe how users perform certain tasks and learn their preferences. Since these agents are meant to be assistive, rather than autonomous, they are able to interact with the user to get additional information or verify if an action should be performed. Similar to conversational recommender systems, learning interface agents are designed to assist with one specific task. In contrast, our agent does not know in advance the specific task it will be performing so it can not bias itself toward learning preferences for that task.

Preference-based planning [22] involves incorporating user preferences into automated planning tasks. These preferences are usually defined a priori but there has also been work to learn the planning preferences [23]. This approach learns the probability of a user performing actions based on the previous plans that user has generated. In our work, that would be the equivalent of having the operator control the robot and provide numerous demonstrations of the task before the agent attempted it. Such an approach would not be practical if there were time constrains or if the operator did not have a fully constructed plan for how to perform the task. For example, the operator might have preferences for how a robot should move between locations but be unaware of a route the robot should take.

Our work also has similarities to other areas of learning during human-robot interactions. When a robot learns from a human, it is often beneficial for the robot to understand the environment from the perspective of the human [24]. Breazeal et al. [25] have examined how a robot can learn from a cooperative human teacher by mapping its sensory inputs to how it estimates the human is viewing the environment. This allows the robot to learn from the viewpoint of the teacher and possibly discover information it would not have noticed from its own viewpoint. This is similar to our own work since it involves inferring information about the reasoning of a human. However, it differs in that it involves observing a teacher demonstrate a specific task and learning from that demonstration. In our work, the human does not demonstrate or teach the learning agent (it only corrects the agent's behavior).

6 Discussion

We have shown that our agent is successfully able to adapt its behavior based on an estimate of user trust, but several key areas of future work still exist. Our evaluation focused on a single movement scenario. In the future we would like to examine additional increasingly complex scenarios. These scenarios will involve more complex tasks being performed by the agent and allow more components of the behavior to be modified. This would demonstrate that our approach is applicable to a variety of scenarios and is not biased to learning an operator's movement preferences.

As the complexity of the examined scenarios increases, it would be advantageous to have a more intelligent agent adaptation algorithm. Our current algorithm often examines a large number of behaviors before finding a trustworthy one. It would be unlikely that a human operator would continue to use the robot if extended periods of untrustworthy behavior occurred so the agent will need to perform more efficient behavior adaptation. One possibility to decrease the number of examined behaviors would be to leverage the results of previous behavior adaptations. The agent could identify if the current operator is similar to one it has previously interacted with and modify its behavior accordingly (e.g., using case-based reasoning [26] to reuse previous adaptations). This would also allow the agent to quickly modify its behavior when switching between tasks. Similarly, it could identify relations among behaviors, find trends, or determine which new behaviors would be most promising to explore. For example, the agent might determine that there is a relation between increasing the speed and the amount of time it takes for a behavior to be labelled as untrustworthy (and guide its behavior search accordingly).

The inverse trust metric we have examined relied on limited interaction with the user and was based exclusively on robot performance. This is beneficial when there are limits on the amount of communication between the operator and the agent (e.g., in time-sensitive situations) but may be missing out on important information that could improve behavior adaptation. In future work, we would like to examine a conversational interaction between the agent and operator that allows an exchange of additional information and improves transparency. This would also allow other aspects of human-robot trust to be included in the inverse trust metric.

The inverse trust estimate makes an assumption that the operator may be undertrusting the agent so trust should be increased. It does not take into account situations where overtrust may be occurring. To account for this, the agent could continue to evaluate trust even when it has determined a behavior to be trustworthy and occasionally perform an untrustworthy test behavior. If the operator does not interrupt the agent, the agent could inform the operator that overtrust is occurring.

When interruptions occur, our inverse trust metric assumes that they were a result of robot performance. Instead, they could be a result of the operator changing its goals and interrupting the agent so a new task can be performed. In our future work, we would like to examine how the agent can reason about the

current goals and modify its behavior accordingly [27]. In these situations, the agent should not conclude it was acting in an untrustworthy manner. Instead, the agent could change its behavior to match the new goals.

7 Conclusions

In this paper we have presented an agent that is able to adapt its behavior in an attempt to behave in a manner that promotes trust from its operator. The agent uses an inverse trust estimate based on how often it completes tasks or is interrupted and uses that metric to determine how much trust the operator has in it. The agent can then use this trust estimate to conclude its behavior is trustworthy or modify its behavior if it is untrustworthy.

We evaluated our approach in a simulated environment that involved controlling a robot to move between two locations. We examined three types of operators and found that the agent was able to modify its behavior to what we would expect to be trustworthy for each operator type. Additionally, we found that there were some behaviors that seem trustworthy but were actually not. The agent was able to identify such behaviors and adapt accordingly.

Acknowledgments

Thanks to the United States Naval Research Laboratory and the Office of Naval Research for supporting this research.

References

1. Desai, M., Medvedev, M.S., Vázquez, M., McSheehy, S., Gadea-Omelchenko, S., Bruggeman, C., Steinfeld, A., Yanco, H.A.: Effects of changing reliability on trust of robot systems. In: 7th ACM/IEEE International Conference on Human-Robot Interaction. (2012) 73–80
2. Desai, M., Kaniarasu, P., Medvedev, M., Steinfeld, A., Yanco, H.: Impact of robot failures and feedback on real-time trust. In: 8th ACM/IEEE International Conference on Human-robot Interaction. (2013) 251–258
3. Sabater, J., Sierra, C.: Review on computational trust and reputation models. Artificial Intelligence Review **24**(1) (2005) 33–60
4. Esfandiari, B., Chandrasekharan, S.: On how agents make friends: Mechanisms for trust acquisition. In: 4th Workshop on Deception, Fraud and Trust in Agent Societies. (2001) 27–34
5. Kaniarasu, P., Steinfeld, A., Desai, M., Yanco, H.A.: Robot confidence and trust alignment. In: 8th ACM/IEEE International Conference on Human-Robot Interaction. (2013) 155–156
6. Jian, J.Y., Bisantz, A.M., Drury, C.G.: Foundations for an empirically determined scale of trust in automated systems. International Journal of Cognitive Ergonomics **4**(1) (2000) 53–71
7. Muir, B.M.: Trust between humans and machines, and the design of decision aids. International Journal of Man-Machine Studies **27**(56) (1987) 527–539

8. Oleson, K.E., Billings, D.R., Kocsis, V., Chen, J.Y., Hancock, P.A.: Antecedents of trust in human-robot collaborations. In: 1st International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support. (2011) 175–178
9. Hancock, P.A., Billings, D.R., Schaefer, K.E., Chen, J.Y., De Visser, E.J., Parasuraman, R.: A meta-analysis of factors affecting trust in human-robot interaction. Human Factors: The Journal of the Human Factors and Ergonomics Society **53**(5) (2011) 517–527
10. Kaniarasu, P., Steinfeld, A., Desai, M., Yanco, H.A.: Potential measures for detecting trust changes. In: 7th ACM/IEEE International Conference on Human-Robot Interaction. (2012) 241–242
11. Trafton, J.G., Monk, C.A.: Task interruptions. Reviews of Human Factors and Ergonomics **3**(1) (2007) 111–126
12. Knexus Research Corporation: eBotworks. `http://www.knexusresearch.com/products/ebotworks.php` (2013) [Online; accessed January 14, 2014].
13. Khasawneh, M.T., Bowling, S.R., Jiang, X., Gramopadhye, A.K., Melloy, B.J.: A model for predicting human trust in automated systems. In: International Conference on Industrial Engineering Theory, Applications and Practice. (2003) 216–222
14. Saleh, J.A., Karray, F., Morckos, M.: Modelling of robot attention demand in human-robot interaction using finite fuzzy state automata. In: IEEE International Conference on Fuzzy Systems. (2012) 1–8
15. Shapiro, D., Shachter, R.: User-agent value alignment. In: Stanford Spring Symposium - Workshop on Safe Learning Agents. (2002)
16. McGinty, L., Smyth, B.: On the role of diversity in conversational recommender systems. In: 5th International Conference on Case-Based Reasoning. (2003) 276–290
17. Mahmood, T., Ricci, F.: Improving recommender systems with adaptive conversational strategies. In: 20th ACM Conference on Hypertext and Hypermedia. (2009) 73–82
18. Chen, K., Zhang, Y., Zheng, Z., Zha, H., Sun, G.: Adapting ranking functions to user preference. In: 24th International Conference on Data Engineering Workshops. (2008) 580–587
19. Maes, P., Kozierok, R.: Learning interface agents. In: 11th National Conference on Artificial Intelligence. (1993) 459–465
20. Horvitz, E.: Principles of mixed-initiative user interfaces. In: 18th Conference on Human Factors in Computing Systems. (1999) 159–166
21. Schlimmer, J.C., Hermens, L.A.: Software agents: Completing patterns and constructing user interfaces. Journal of Artificial Intelligence Research **1** (1993) 61–89
22. Baier, J.A., McIlraith, S.A.: Planning with preferences. AI Magazine **29**(4) (2008) 25–36
23. Li, N., Kambhampati, S., Yoon, S.W.: Learning probabilistic hierarchical task networks to capture user preferences. In: 21st International Joint Conference on Artificial Intelligence. (2009) 1754–1759
24. Berlin, M., Gray, J., Thomaz, A.L., Breazeal, C.: Perspective taking: An organizing principle for learning in human-robot interaction. In: 21st National Conference on Artificial Intelligence. (2006) 1444–1450
25. Breazeal, C., Gray, J., Berlin, M.: An embodied cognition approach to mindreading skills for socially intelligent robots. International Journal of Robotic Research **28**(5) (2009)
26. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications **7**(1) (1994) 39–59

27. Klenk, M., Molineaux, M., Aha, D.W.: Goal-driven autonomy for responding to unexpected events in strategy simulations. Computational Intelligence **29**(2) (2013) 187–206