How Much Do You Trust Me? Learning a Case-Based Model of Inverse Trust

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

[1] Knexus Research Corporation; Springfield, VA; USA
[2] Navy Center for Applied Research in Artificial Intelligence; Naval Research Laboratory (Code 5514); Washington, DC; USA
{*first.last*}@knexusresearch.com | david.aha@nrl.navy.mil

Abstract. Robots can be important additions to human teams if they improve team performance by providing new skills or improving existing skills. However, to get the full benefits of a robot the team must trust and use it appropriately. We present an agent algorithm that allows a robot to estimate its trustworthiness and adapt its behavior in an attempt to increase trust. It uses case-based reasoning to store previous behavior adaptations and uses this information to perform future adaptations. We compare case-based behavior adaptation to behavior adaptation that does not learn and show it significantly reduces the number of behaviors that need to be evaluated before a trustworthy behavior is found. Our evaluation is in a simulated robotics environment and involves a movement scenario and a patrolling/threat detection scenario.

1 Introduction

Robots can be important members of human teams if they provide capabilities that are critical for accomplishing team goals and complement those of their human teammates. These could include improved sensory capabilities, communication capabilities, or an ability to operate in environments humans can not (e.g., rough terrain or dangerous situations). Including these robots might be necessary for the team to meet its objectives and reduce human risk. However, to make full use of these robots the human teammates will need to trust them.

This is especially important for robots that operate autonomously or semi-autonomously. In these situations, their human operator(s) would likely issue commands or delegate tasks to the robot to reduce their workload or more efficiently achieve team goals. A lack of trust in the robot could result in the humans under-utilizing it, unnecessarily monitoring the robot's actions, or possibly not using it at all [1].

A robot could be designed so that it operates in a sufficiently trustworthy manner. However, this may be impractical because the measure of trust might be task-dependent, user-dependent, or change over time [2]. For example, if a robot receives a command from an operator to navigate between two locations in a city, one operator might prefer the task be performed as quickly as possible whereas another might prefer the task be performed as safely as possible

(e.g., not driving down a road with heavy automobile traffic or large potholes). Each operator has distinct preferences that influence how they will trust the robot's behavior, and these preferences may conflict. Even if these user preferences were known in advance, a change in context could also influence what behaviors are trustworthy. An operator who generally prefers a task to be performed quickly would likely change that preference if the robot was transporting hazardous material, whereas an operator who prefers safety would likely change their preferences in an emergency situation. Similarly, it may be infeasible to elicit a complete knowledge base of rules defining trustworthy behavior if the experts do not know the explicit rules or there are so many rules it is impractical to extract them all.

The ability of a robot to behave in a trustworthy manner regardless of the operator, task, or context requires that it can evaluate its trustworthiness and adapt its behavior accordingly. The robot may not always get explicit feedback about its trustworthiness but will instead need to estimate its trustworthiness based on its interactions with its operator. Such an estimate, which we refer to as an *inverse trust estimate*, differs from traditional computational trust metrics in that it measures how much trust another agent has in the robot rather than how much trust the robot has in another agent. In this paper we examine how a robot can estimate the trust an operator has in it, adapt its behavior to become more trustworthy, and learn from previous adaptations so it can perform trustworthy behaviors more quickly. We use case-based reasoning (CBR) to allow the robot to learn from previous behavior adaptations. The robot stores previous behavior adaptation information as cases in its case base and uses those cases to perform future behavior adaptations.

In the remainder of this paper we describe our behavior adaptation approach and evaluate it in a simulated robotics domain. We describe the robot's behavior and the aspects that it can modify in Section 2. Section 3 presents the inverse trust metric and Section 4 describes how it can be used to guide the robot's behavior. In Section 5, we evaluate our case-based behavior adaptation strategy in a simulated robotics domain and report evidence that it can efficiently adapt the robot's behavior to the operator's preferences. Related work is examined in Section 6 followed by a discussion of future work and concluding remarks in Section 7.

2 Agent Behavior

We assume the robot can control and modify aspects of its behavior. These modifiable components could include changing a module (e.g., switching between two path planning algorithms), its parameter values, or its data (e.g., using a different map of the environment). By modifying these components the robot can immediately change its behavior.

We define each modifiable behavior component i to have a range of selectable values $\mathcal{C}_i$. If the robot has m modifiable components, its current behavior B

will be a tuple containing the currently selected value c_i for each modifiable component ($c_i \in \mathcal{C}_i$):

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

By changing one or more of its behavior components, the robot switches from using its current behavior B to a new behavior $B^{'}$. While operating in the environment, the robot might change its behavior several times, resulting in a sequence of behaviors $\langle B_1, B_2, \ldots, B_n \rangle$. Since the goal of the robot is to perform trustworthy behavior, behavior changes will occur because a current behavior B was found to be untrustworthy and it is attempting to perform a more trustworthy behavior.

3 Inverse Trust Estimate

Traditional trust metrics are used to estimate the trust an agent should have in other agents [3]. The agent can use prior interactions with those agents or feedback from others to determine their trustworthiness. The information this agent uses is likely internal to it and not directly observable by a third party. In a robotics context, the robot will not be able to observe the information a human operator uses to assess their trust in it. Instead, the robot will need to acquire this internal information to estimate operator trust.

One option would be to directly ask the operator, either as it is interacting with the robot [4] or after the task has been completed [5, 6], about how trustworthy the robot was behaving. However, this might not be practical in situations that are time-sensitive or where there would be a significant delay between when the robot wishes to evaluate its trustworthiness and the next opportunity to ask the operator (e.g., during a multi-day search and rescue mission). An alternative that does not require direct operator feedback is for the robot to *infer* the trust the operator has in it.

Factors that influence human-robot trust can be grouped into three main categories [1]: 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 type). Although these factors have all been shown to influence human-robot trust, the strongest indicator of trust is robot performance [7, 8]. Kaniarasu et al. [9] have used an inverse trust metric that estimates robot performance based on the number of times the operator warns the robot about its behavior and the number of times the operator takes manual control of the robot. They found this metric aligns closely with the results of trust surveys performed by the operators. However, this metric does not take into account factors of the robot's behavior that increase trust.

The inverse trust metric we use is based on the number of times the robot completes an assigned task, fails to complete a task, or is interrupted while performing a task. An interruption occurs when the operator instructs the robot to stop its current autonomous behavior. Our robot infers that any interruptions are a result of the operator being unsatisfied with the robot's performance.

Similarly, our robot assumes the operator will be unsatisfied with any failures and satisfied with any completed tasks. Interrupts could also be a result of a change in the operator's goals, or failures could be a result of unachievable tasks, but the robot works under the assumption that those situations occur rarely.

Our control strategy estimates whether trust is increasing, decreasing, or remaining constant while the current behavior $B^{'}$ is being used by the robot. We estimate this value as follows:

$$Trust_{B'} = \sum_{i=1}^{n} w_i \times cmd_i,$$

where there were n commands issued to the robot while it was using its current behavioral configuration. If the ith command ($1 \leq i \leq n$) was interrupted or failed it will decrease the trust value and if it was completed successfully it will increase the trust value ($cmd_i \in \{-1, 1\}$). The ith command will also receive a weight ($w_i = [0, 1]$) related to the command (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).

4 Trust-guided Behavior Adaptation Using CBR

The robot uses the inverse trust estimate to infer if its current behavior is trustworthy, is not trustworthy, or it does not yet know. We use two threshold values to identify trustworthy and untrustworthy behavior: the trustworthy threshold (τ_T) and the untrustworthy threshold (τ_{UT}). Our robot uses the following tests:

- If the trust value reaches the trustworthy threshold ($Trust_{B'} \geq \tau_T$), the robot will conclude it has found a sufficiently trustworthy behavior (although it may continue evaluating trust in case any changes occur).
- If the trust value falls to or below the untrustworthy threshold ($Trust_{B'} \leq \tau_{UT}$), the robot will modify its behavior in an attempt to be more trustworthy.
- If the trust value is between the two thresholds ($\tau_{UT} < Trust_{B'} < \tau_T$), the robot will continue to evaluate the operator's trust.

In the situations where the trustworthy threshold has been reached or neither threshold has been reached, the robot will continue to use its current behavior. However, when the untrustworthy threshold has been reached the robot will modify its behavior in an attempt to behave in a more trustworthy manner.

When a behavior B is found by the robot to be untrustworthy it is stored as an evaluated pair E that also contains the time t it took the behavior to be labeled as untrustworthy:

$$E = \langle B, t \rangle$$

The time it took for a behavior to reach the untrustworthy threshold is used to compare behaviors that have been found to be untrustworthy. A behavior $B^{'}$

that reaches the untrustworthy threshold more quickly than another behavior $B^{''}$ ($t^{'} < t^{''}$) is assumed to be less trustworthy than the other. This is based on the assumption that if a behavior took longer to reach the untrustworthy threshold then it was likely performing some trustworthy actions or was not performing untrustworthy actions as quickly.

As the robot evaluates behaviors, it stores a set $\mathcal{E}_{past}$ of previously evaluated behaviors ($\mathcal{E}_{past} = \{E_1, E_2, \ldots, E_n\}$). It continues to add to this set until it locates a trustworthy behavior B_{final} (when the trustworthy threshold is reached), if a trustworthy behavior exists. The sets of evaluated behaviors can be thought of as the search path that resulted in the final solution (the trustworthy behavior). The search path information is potentially useful because if the robot can determine it is on a similar search path that it has previously encountered (similar behaviors being labeled untrustworthy in a similar amount of time) then the robot can identify what final behavior it should attempt.

To allow for the reuse of past behavior adaptation information we use case-based reasoning. Each *case* C is composed of a problem and a solution. In our context, the *problem* is the previously evaluated behaviors and the *solution* is the final trustworthy behavior:

$$C = \langle \mathcal{E}_{past}, B_{final} \rangle$$

These cases are stored in a *case base* and represent the robot's knowledge about previous behavior adaptation.

When the robot modifies its behavior it selects new values for one or more of the modifiable components. The new behavior B_{new} is selected as a function of all behaviors that have been previously evaluated for this operator and its case base CB:

$$B_{new} = selectBehavior(\mathcal{E}_{past}, CB)$$

The *selectBehavior* function (Algorithm 1) attempts to use previous adaptation experience to guide the current adaptation. The algorithm iterates through each case in the case base (line 2) and checks to see if that case's final behavior has already been evaluated (line 3). If so, the robot has already found the behavior to be untrustworthy and does not try to use it again. Algorithm 1 then compares the sets of evaluated behaviors of the remaining cases ($C_i.\mathcal{E}_{past}$) to the robot's current set of evaluated behaviors ($\mathcal{E}_{past}$) using a similarity metric (line 4). The most similar case's final behavior is returned and will be used by the robot (line 10). If no such behaviors are found (the final behaviors of all cases have been examined or the case base is empty), the $modifyBehavior$ function is used to select the next behavior to perform (line 9). It selects an evaluated behavior E_{max} that took the longest to reach the untrustworthy threshold ($\forall E_i \in \mathcal{E}_{past}(E_{max}.t \geq E_i.t)$) and performs a random walk (without repetition) to find a behavior B_{new} that required the minimum number of changes from $E_{max}.B$ and has not already been evaluated ($\forall E_i \in \mathcal{E}_{past}(B_{new} \neq E_i.B)$). If all possible behaviors have been evaluated and found to be untrustworthy the robot will stop adapting its behavior and use the behavior from E_{max}.

Algorithm 1: Selecting a New Behavior

Function: *selectBehavior*($\mathcal{E}_{past}$, CB) ***returns*** B_{new};

```
1  bestSim ← 0; B_best ← ∅;
2  foreach C_i ∈ CB do
3      if C_i.B_final ∉ E_past then
4          sim_i ← sim(E_past, C_i.E_past);
5          if sim_i > bestSim then
6              bestSim ← sim_i;
7              B_best ← C_i.B_final;
8  if B_best = ∅ then
9      B_best ← modifyBehavior(E_past);
10 return B_best;
```

The similarity between two sets of evaluated behaviors (Algorithm 2) is complicated by the fact that the sets may vary in size. The size of the sets depends on the number of previous behaviors that were evaluated by the robot in each set and there is no guarantee that the sets contain identical behaviors. To account for this, the similarity function looks at the overlap between the two sets and ignores behaviors that have been examined in only one of the sets. Each evaluated behavior in the first set is matched to an evaluated behavior E_{max} in the second set that contains the most similar behavior (line 3, $sim(B_1, B_2) = \frac{1}{m}\sum_{i=1}^{m} sim(B_1.c_i, B_2.c_i)$, where the similarity function will depend on the specific type of behavior component). If those behaviors are similar enough, based on a threshold λ (line 4), then the similarity of the time components of these evaluated behaviors are included in the similarity calculation (line 5). This ensures that only matches between evaluated behaviors that are highly similar (i.e., similar behaviors exist in both sets) are included in the similarity calculation (line 9). The similarity metric only includes comparisons between time components because the goal is to find when similar behaviors were found to be untrustworthy in a similar amount of time.

5 Evaluation

In this section, we describe an evaluation for our claim that our case-based reasoning approach can adapt, identify, and perform trustworthy behaviors more quickly than a random walk approach. We conducted this study in a simulated environment with a simulated robot and operator. We examined two robotics scenarios: movement and patrolling for threats.

5.1 eBotWorks Simulator

Our evaluation uses the eBotworks simulation environment [10]. eBotworks is a multi-agent simulation engine and testbed that allows for multimodal command

Algorithm 2: Similarity between sets of evaluated behaviors

Function: *sim*($\mathcal{E}_1$, $\mathcal{E}_2$) ***returns*** *sim*;

$totalSim \leftarrow 0$; $num \leftarrow 0$;
foreach $E_i \in \mathcal{E}_1$ **do**
    $E_{max} \leftarrow \arg\max_{E_j \in \mathcal{E}_2} (sim(E_i.B, E_j.B))$;
    if $sim(E_i.B, E_{max}.B) > \lambda$ **then**
        $totalSim \leftarrow totalSim + sim(E_i.t, E_{max}.t)$;
        $num \leftarrow num + 1$;
if $num = 0$ **then**
    return 0;
return $\frac{totalSim}{num}$;

and control of unmanned systems. It allows for autonomous agents to control simulated robotic vehicles while interacting with human operators, and for the autonomous behavior to be observed and evaluated. We chose to use eBotworks based on its flexibility in autonomous behavior modeling, the ability for agents to process natural language commands, and built-in experimentation and data collection capabilities.

The robot operates in a simulated urban environment containing landmarks (e.g., roads) and objects (e.g., houses, humans, traffic cones, vehicles, road barriers). The robot is 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 a robot in eBotworks are non-deterministic (e.g., the robot cannot anticipate its exact position after moving).

5.2 Experimental Conditions

We use simulated operators in our study to issue commands to the robot. In each experiment, one of these operators interacts with the robot for 500 *trials*. The simulated operators differ in their preferences, which will influence how they evaluate the robot's performance (when an operator allows the robot to complete a task and when it interrupts). At the start of each trial, the robot randomly selects (with a uniform distribution) initial values for each of its modifiable behavior components. Throughout the trial, a series of experimental *runs* will occur. Each run involves the simulated operator issuing a command to the robot and monitoring the robot as it performs the assigned task. During a run, the robot might complete the task, fail to complete the task, or be interrupted by the operator. At the end of a run the environment will be reset so a new run can begin. The results of these runs will be used by the robot to estimate the operator's trust in it and to adapt its behavior if necessary. A trial concludes

when the robot successfully identifies a trustworthy behavior or it has evaluated all possible behaviors.

For the case-based behavior adaptation, at the start of each experiment the robot will have an empty case base. At the end of any trial where the robot has found a trustworthy behavior and has performed at least one random walk adaptation (i.e., the agent could not find a solution by only using information in the case base), a case will be added to the case base and can be used by the robot in subsequent trials. The added case represents the trustworthy behavior found by the robot and the set of untrustworthy behaviors that were evaluated before the trustworthy behavior was found.

We set the robot's trustworthy threshold $\tau_T = 5.0$ and its untrustworthy threshold $\tau_{UT} = -5.0$. These threshold values were chosen to allow some fluctuation between increasing and decreasing trust while still identifying trustworthy and untrustworthy behaviors quickly. To calculate the similarity between sets of evaluated behaviors we set the similarity threshold to be $\lambda = 0.95$ (behaviors must be 95% similar to be matched). This threshold was used so that only highly similar behaviors will be matched together.

5.3 Scenarios

The scenarios we evaluate, movement and patrolling for threats, were selected to demonstrate the ability of our behavior adaptation technique when performing increasingly complex tasks. While the movement scenario is fairly simple, the patrolling scenario involves a more complex behavior with more modifiable behavior components.

Movement Scenario: The initial task the robot is required to perform involves moving between two locations in the environment. The simulated operators used in this scenario assess their trust in the robot using three performance metrics:

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

We use three simulated operators:

- **Speed-focused operator**: This operator prefers the robot to move to the destination quickly regardless of whether it hits any obstacles ($t_{complete} = 15$ seconds, $p_\alpha = 95\%$, $p_\gamma = 5\%$).
- **Safety-focused operator**: This operator prefers the robot 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**: This operator prefers a balanced mixture of speed and safety ($t_{complete} = 15$ seconds, $p_\alpha = 95\%$, $p_\gamma = 95\%$).

The robot has two modifiable behavior components: *speed* (meters per second) and *obstacle padding* (meters). Speed relates to how fast the robot can move and obstacle padding relates to the distance the robot will attempt to maintain from obstacles during movement. The set of possible values for each modifiable component ($\mathcal{C}_{speed}$ and $\mathcal{C}_{padding}$) are determined from minimum and maximum values (based on the robot's capabilities) with fixed increments.

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

Patrolling Scenario: The second task the robot is required to perform involves patrolling between two locations in the environment. At the start of each run, 6 suspicious objects representing potential threats are randomly placed in the environment. Of those 6 suspicious objects, between 0 and 3 (inclusive) denote hazardous explosive devices (selected randomly using a uniform distribution). As the robot moves between the start location and the destination it will scan for suspicious objects nearby. When it identifies a suspicious object it will pause its patrolling behavior, move toward the suspicious object, scan it with its explosives detector, label the object as an explosive or harmless, and then continue its patrolling behavior. The accuracy of the explosives detector the robot uses is a function of how long the robot spends scanning the object (longer scan times result in improved accuracy) and its proximity to the object (smaller scan distances increase the accuracy). The scan time (seconds) and scan distance (meters) are two modifiable components of the robot's behavior whose set of possible values are:

$$\mathcal{C}_{scantime} = \{0.5, 1.0, \ldots, 5.0\}$$
$$\mathcal{C}_{scandistance} = \{0.25, 0.5, \ldots, 1.0\}$$

The simulated operators in this scenario base their decision to interrupt the robot on its ability to successfully identify suspicious objects and label them correctly (in addition to the task duration, task completion, and safety factors discussed in the movement scenario). An operator will interrupt the robot if it does not scan one or more of the suspicious objects or incorrectly labels a harmless object as an explosive. In the event that the robot incorrectly labels an explosive device as harmless, the explosive will eventually detonate and the robot will fail its task. When determining its trustworthiness, the robot will give higher weights to failures due to missing explosive devices (they will be weighted 3 times higher than other failures or interruptions).

In this scenario we use two simulated operators:

– **Speed-focused operator**: The operator prefers that the robot performs the patrol task within a fixed time limit ($t_{complete} = 120$ seconds, $p_\alpha = 95\%$, $p_\gamma = 5\%$).

– **Detection-focused operator**: The operator prefers the task be performed correctly regardless of time ($t_{complete} = 120$ seconds, $p_\alpha = 5\%$, $p_\gamma = 5\%$).

5.4 Results

We found that both the case-based behavior adaptation and the random walk behavior adaptation strategies resulted in similar trustworthy behaviors for each simulated operator. In the movement scenario, for the speed-focused operator the trustworthy behaviors had higher speeds regardless of padding ($3.5 \leq speed \leq 10.0$, $0.1 \leq padding \leq 1.9$). The safety-focused operator had higher padding regardless of speed ($0.5 \leq speed \leq 10.0$, $0.4 \leq padding \leq 1.9$). Finally, the balanced operator had higher speed and higher padding ($3.5 \leq speed \leq 10.0$, $0.4 \leq padding \leq 1.9$). These results are consistent with our previous findings [11] that trust-guided behavior adaptation using random walk converges to behaviors that appear to be trustworthy for each type of operator.

In the patrolling scenario, which we have not studied previously, the differences between the trustworthy behaviors for the two operators are not only in the ranges of the values for the modifiable components but also their relations to each other. Similar to what was seen in the movement scenario, since the speed-focused patrol operator has a time preference the robot only converges to higher speed values whereas the detection-focused operator has no such restriction (the speed-focused operator never converges to a speed below 2.0). The speed-focused patrol operator never has both a low speed and a high scan time. This is because these modifiable components are interdependent. If the robot spends more time scanning, it will need to move through the environment at a higher speed. Similarly, both operators converge to scan time and scan distance values that reveal a dependence. The robot only selects a poor value for one of the modifiable components (low scan time or high scan distance) if it selects a very good value for the other component (high scan time or low scan distance). This shows that behavior adaptation can select trustworthy values when the modifiable components are mostly independent or when there is a strong dependence between multiple behavior components.

Both the case-based reasoning and random walk adaptation approaches converged to similar trustworthy behaviors. The only noticeable difference is that final behaviors stored in cases are found to be trustworthy in more trials. This is what we would expect from the case-based approach since these cases are retrieved and their final behaviors are reused. The primary difference between the two behavior adaption approaches was related to the number of behaviors that needed to be evaluated before a trustworthy behavior was found. Table 1 shows the mean number of evaluated behaviors (and 95% confidence interval) when interacting with each operator type (over 500 trials for each operator). The table also lists the number of cases acquired during the case-based behavior adaptation experiments (each experiment started with an empty case base). In addition to being controlled by only a single operator, we also examined a condition in which, for each scenario, the operator is selected at random with equal probability. This represents a more realistic scenario where the robot will

be required to interact with a variety of operators without any knowledge about which operator will control it.

Table 1. Mean number of behaviors evaluated before finding a trustworthy behavior

Scenario	Operator	Random Walk	Case-based	Cases Acquired
Movement	*Speed-focused*	20.3 (±3.4)	1.6 (±0.2)	24
Movement	*Safety-focused*	2.8 (±0.3)	1.3 (±0.1)	18
Movement	*Balanced*	27.0 (±3.8)	1.8 (±0.2)	33
Movement	*Random*	14.6 (±2.9)	1.6 (±0.1)	33
Patrol	*Speed-focused*	344.5 (±31.5)	9.9 (±3.9)	25
Patrol	*Detection-focused*	199.9 (±23.3)	5.5 (±2.2)	22
Patrol	*Random*	269.0 (±27.1)	9.3 (±3.2)	25

The case-based approach required significantly fewer behaviors to be evaluated in all seven experiments (using a paired t-test with $p < 0.01$). This is because the case-based approach could learn from previous adaptations and use that information to quickly find trustworthy behaviors. At the beginning of a trial, when the robot's case base is empty, the case-based approach must perform adaptation that is similar to the random walk approach. As the case base size grows, the number of times random walk adaptation is required decreases until the agent generally performs only one case-based behavior adaptation before finding a trustworthy behavior. Even when the case base contains cases from all two (in the patrol scenario) or three (in the movement scenario) simulated operators, the case-based approach can quickly differentiate between the users and select a trustworthy behavior for the current operator. The number of adaptations required for the safety-focused and detection-focused operators were lower than for the other operators in their scenarios because a higher percentage of behaviors are considered trustworthy for those operators.

5.5 Discussion

The primary limitation of the case-based approach is that it relies on the random walk search when it does not have any suitable cases to use. Although the mean number of behaviors evaluated by the case-based approach is low, the situations where random walk is used require an above-average number of behaviors to be evaluated (closer to the mean number of behaviors evaluated when only random walk is used). For example, if we consider only the final 250 trials for each of the patrol scenario operators the mean number of behaviors evaluated is lower than the overall mean (4.2 for the speed-focused, 2.8 for the detection-focused, and 3.3 for the random). This is because the robot performs the more expensive random walk adaptations in the early trials and generates cases that are used in subsequent trials.

Two primary solutions exist to reduce the number of behaviors examined: improved search and seeding of the case base. We used random walk search be-

cause it requires no explicit knowledge about the domain or the task. However, a more intelligent search that could identify relations between interruptions and modifiable components (e.g., an interruption when the robot is close to objects requires a change to the padding value) would likely improve adaptation time. Since a higher number of behaviors need to be evaluated when new cases are created, if a set of initial cases were provided to the robot it would be able to decrease the number of random walk adaptations (or adaptations requiring a different search technique) it would need to perform. These two solutions introduce their own potential limitations. A more informed search requires introducing domain knowledge, which may not be easy to obtain, and seeding the case base requires an expert to manually author cases (or another method for case acquisition). The specific requirements of the application domain will influence whether faster behavior adaptation or lower domain knowledge requirements are more important.

6 Related Work

In addition to Kaniarasu et al. [9], Saleh et al. [12] have also proposed a measure of inverse trust and use a set of expert-authored rules to measure trust. Unlike our own work, while these approaches measure trust, they do not use this information to adapt behavior. The limitation of using these trust metrics to guide our behavior adaptation technique is that one of the metrics only measures decreases in trust [9] and the other requires expert-authored rules [12].

The topic of trust models in CBR is generally examined in the context of recommendation systems [13] or agent collaboration [14]. Similarly, the idea of case provenance [15] is related to trust in that it involves considering the source of a case and if that source is a reliable source of information. These investigations consider traditional trust, where an agent determines its trust in another agent, rather than inverse trust, which is our focus.

Case-based reasoning has been used for a variety of robotics applications and often facilitates action selection [16] or behavior selection [17]. Existing work on CBR and robotics differs from our own in that most systems attempt to optimize the robot's performance without considering that sub-optimal performance may be necessary to gain a human teammate's trust. In an assistive robotics task, a robotic wheelchair uses CBR to learn to drive in a similar manner to its operator [18]. This work differs from our own in that it requires the operator to demonstrate the behavior over several trials, like a teacher, and the robot learns from those observations. The robot in our system received information that is not annotated so it can not benefit from direct feedback or labelling by the operator.

Shapiro and Shachter [19] discuss the need for an agent to act in the best interests of a user even if that requires sub-optimal performance. Their work is on identifying factors that influence the user's utility function and updating the agent's reward function accordingly. This is similar to our own work in that

behavior is modified to align with a user's preference, but our robot is not given an explicit model of the user's reasoning process.

Conversational recommender systems [20] iteratively improve recommendations to a user by tailoring the recommendations to the user's preferences. As more information is obtained through dialogs with a user, these systems refine their model of that user. Similarly, learning interface agents observe a user performing a task (e.g., sorting e-mail [21] or schedule management [22]) and learn the user's preferences. Both conversational recommender systems and learning interface agents are designed to learn preferences for a single task whereas our behavior adaptation requires no prior knowledge about what tasks will be performed.

Our work also relates 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 that human. Breazeal et al. [23] 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 preference-based planning systems that learn a user's preferences for plan generation [24]. Like our own work, these systems involve inferring information about the reasoning of a human. However, they differ in that they involve observing a teacher demonstrate a specific task and learning from those demonstrations.

7 Conclusions

In this paper we presented an inverse trust measure that allows a robot to estimate an operator's trust and adapt its behavior to increase trust. As the robot performs trust-guided adaptation, it learns using case-based reasoning. Each time it successfully finds a trustworthy behavior, it can record a case that contains the trustworthy behavior as well as the sequence of untrustworthy behaviors that it evaluated.

We evaluated our trust-guided behavior adaptation algorithm in a simulated robotics environment by comparing it to a behavior adaptation algorithm that does not learn from previous adaptations. Two scenarios were examined: movement and patrolling for threats. Both approaches converge to trustworthy behaviors for each type of operator but the case-based algorithm requires significantly fewer behaviors to be evaluated before a trustworthy behavior is found. This is advantageous because the chances that the operator will stop using the robot increase the longer the robot is behaving in an untrustworthy manner.

Although we have shown the benefits of trust-guided behavior adaptation, several areas of future work exist. In longer scenarios it may be important to not only consider undertrust, as we have done in this work, but also overtrust. In situations of overtrust, the operator may trust the robot too much and allow the robot to behave autonomously even when it is performing poorly. We also

plan to include additional trust factors in the inverse trust estimate and add mechanisms that promote transparency between the robot and operator. More generally, adding an ability for the robot to reason about its own goals and the goals of the operator would allow the robot to verify it is trying to achieve the same goals as the operator and identify any unexpected goal changes (e.g., such as when a threat occurs). Examining more complex interactions between the operator and the robot, like providing preferences or explaining the reasons for interruptions, would allow the robot to build a more elaborate operator model and potentially use different learning strategies than those presented here.

Acknowledgments

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

References

1. 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
2. Desai, M., Kaniarasu, P., Medvedev, M., Steinfeld, A., Yanco, H.: Impact of robot failures and feedback on real-time trust. In: 8th 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. Kaniarasu, P., Steinfeld, A., Desai, M., Yanco, H.A.: Robot confidence and trust alignment. In: 8th International Conference on Human-Robot Interaction. (2013) 155–156
5. 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
6. 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
7. 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
8. Carlson, M.S., Desai, M., Drury, J.L., Kwak, H., Yanco, H.A.: Identifying factors that influence trust in automated cars and medical diagnosis systems. In: AAAI Symposium on The Intersection of Robust Intelligence and Trust in Autonomous Systems. (2014) 20–27
9. Kaniarasu, P., Steinfeld, A., Desai, M., Yanco, H.A.: Potential measures for detecting trust changes. In: 7th International Conference on Human-Robot Interaction. (2012) 241–242
10. Knexus Research Corporation: eBotworks. `http://www.knexusresearch.com/products/ebotworks.php` (2013) [Online; accessed April 9, 2014].

11. Floyd, M.W., Drinkwater, M., Aha, D.W.: Adapting autonomous behavior using an inverse trust estimation. In: 14th International Conference on Computational Science and Its Applications - Workshop on New Trends in Trust Computational Models. (2014)
12. Saleh, J.A., Karray, F., Morckos, M.: Modelling of robot attention demand in human-robot interaction using finite fuzzy state automata. In: International Conference on Fuzzy Systems. (2012) 1–8
13. Tavakolifard, M., Herrmann, P., Öztürk, P.: Analogical trust reasoning. In: 3rd International Conference on Trust Management. (2009) 149–163
14. Briggs, P., Smyth, B.: Provenance, trust, and sharing in peer-to-peer case-based web search. In: 9th European Conference on Case-Based Reasoning. (2008) 89–103
15. Leake, D., Whitehead, M.: Case provenance: The value of remembering case sources. In: 7th International Conference on Case-Based Reasoning. (2007) 194–208
16. Ros, R., Veloso, M.M., de Mántaras, R.L., Sierra, C., Arcos, J.L.: Retrieving and reusing game plays for robot soccer. In: 8th European Conference on Case-Based Reasoning. (2006) 47–61
17. Likhachev, M., Arkin, R.C.: Spatio-temporal case-based reasoning for behavioral selection. In: International Conference on Robotics and Automation. (2001) 1627–1634
18. Urdiales, C., Peula, J.M., Fernández-Carmona, M., Hernández, F.S.: Learning-based adaptation for personalized mobility assistance. In: 21st International Conference on Case-Based Reasoning. (2013) 329–342
19. Shapiro, D., Shachter, R.: User-agent value alignment. In: Stanford Spring Symposium - Workshop on Safe Learning Agents. (2002)
20. McGinty, L., Smyth, B.: On the role of diversity in conversational recommender systems. In: 5th International Conference on Case-Based Reasoning. (2003) 276–290
21. Maes, P., Kozierok, R.: Learning interface agents. In: 11th National Conference on Artificial Intelligence. (1993) 459–465
22. Horvitz, E.: Principles of mixed-initiative user interfaces. In: 18th Conference on Human Factors in Computing Systems. (1999) 159–166
23. 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)
24. 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