S.I.: EXPLAINABLE SEQUENTIAL DECISION-MAKING

Single-shot policy explanation to improve task performance via semantic reward coaching

Aaquib Tabrez[1,2] · Ryan Leonard[1] · Bradley Hayes[1]

Received: 17 June 2024 / Accepted: 3 July 2025

Abstract
Communication is crucial for synchronizing expectations and knowledge within teams. For robots to effectively collaborate with or provide actionable decision-support or coaching to humans, it is critical that they be able to generate intelligible explanations to reconcile differences between their understanding of the world and that of their collaborators. In this work we present Single-shot Policy Elicitation for Augmenting Rewards (SPEAR), a novel sequential optimization algorithm that uses semantic explanations derived from combinations of planning predicates to augment human agents' reward functions, driving their policies to exhibit more optimal behavior by modeling humans as reinforcement learning (RL) agents and reconciling disparities in their reward function. We present an experimental validation of the policy manipulation capabilities of SPEAR in a practically grounded application and a performance analysis of SPEAR across a suite of domains with increasingly complex state spaces and predicate counts. SPEAR demonstrates substantial improvements in runtime and addressable problem size, enabling an expert agent to leverage its own expertise to communicate actionable information to improve human performance. Through a series of human subjects studies, we demonstrate SPEAR's potential to improve human policies and reduce cognitive load, all while enhancing interpretability, task awareness, and promoting active thinking patterns among users. Finally, we apply SPEAR in a robot-to-robot policy manipulation scenario, showcasing its applicability in robot-robot collaborations.

Keywords Human-agent collaboration · Explainable AI · Collaborating robots · Policy explanation

1 Introduction and motivation

Autonomous systems have been shown to improve human performance across a multitude of tasks by imparting useful knowledge or motivating positive behavioral changes [1–3]. As is the case in the domains of human-AI tutoring and coaching, this is accomplished primarily using natural language explanations [4, 5]. Unfortunately, the process of generating concise and informative explanations capable of eliciting desired changes is a difficult task, as it requires both insights into a human collaborator's decision-making process and the ability to determine and convey important information [6–8]. Similarly, for autonomous robots operating with different state representations (e.g., different embodiments or sensors), policy repair requires a common ground (language) to communicate updates for efficient behavior modification during the task. Despite their generalizability, large language models currently fall short in planning and reasoning capabilities and lack real-world grounding, which limits their ability to provide reliable and factual explanations in high-stakes scenarios [9, 10].

In this work, we characterize the problem of semantically manipulating human behavior through external advice, especially when a human teammate displays suboptimal behaviors. We model the human agent as a

https://doi.org/10.1007/s00521-025-11477-y
Published online: 02 August 2025

reinforcement learner, whose suboptimality is attributed to a misguided reward function rather than a faulty policy search algorithm [4]. Our proposed solution empowers autonomous agents to: 1) Infer the reward function driving human agent's behavior; 2) Identify divergences between human reward function and their own; and 3) Offer advice that concisely and efficiently rectifies these differences, enabling human policy repair.

An illustrative scenario we use to demonstrate the importance and practicality of this work is routing people during an emergency building evacuation, where inhabitants are unaware of the precise nature of the emergency (Fig. 1-right). Even if people are capable of navigating toward the nearest exit, uncertainty about the nature of environmental hazards could be disastrous. Adding to the complexity of time-critical, high-consequence scenarios like this one, it is essential that any advice or instructions account for the recipient's knowledge of the world, and therefore must also consider tradeoffs between accuracy, specificity, and interpretability [11, 12]. As an example, someone visiting a building for a meeting may not know how to change their evacuation plan when told "*There's a fire near Conference Room 3,*" but may be able to adapt their plan if told "*the north half of the building is on fire.*" Even though the latter phrase may not communicate the most accurate representation of the hazard, it is more easily comprehensible to someone who is less familiar with the building. Thus, autonomous systems aiming to offer useful feedback must explicitly consider the complexity of their own explanations and the knowledge held by those they attempt to help.

To achieve this, we present **S**ingle-shot **P**olicy **E**licitation for **A**ugmenting **R**ewards (SPEAR), a novel integer programming-based algorithm for generating reward updates in the form of natural language advice to improve the policies of human collaborators. SPEAR enables an autonomous robot to utilize knowledge about the beliefs and goals of its collaborators (whether they are humans or other robotic agents) to identify inaccuracies in their models, generating targeted, interpretable guidance for updating their reward functions (and thus, policies) during a task. Key to its effectiveness, SPEAR generates feedback with levels of specificity appropriate to the human agent's understanding of the world. The four primary contributions of our work are:

- A characterization of the *policy elicitation* problem.
- SPEAR, a novel algorithm for improving task performance through semantic elicitation of others' policies.
- An integer program enabling semantic communication of state space regions that scales linearly with predicate count (an improvement over exponential, exact methods [13]).
- An experimental validation and performance analysis of SPEAR, along with a series of human-subjects studies to validate the utility of SPEAR-generated explanations.

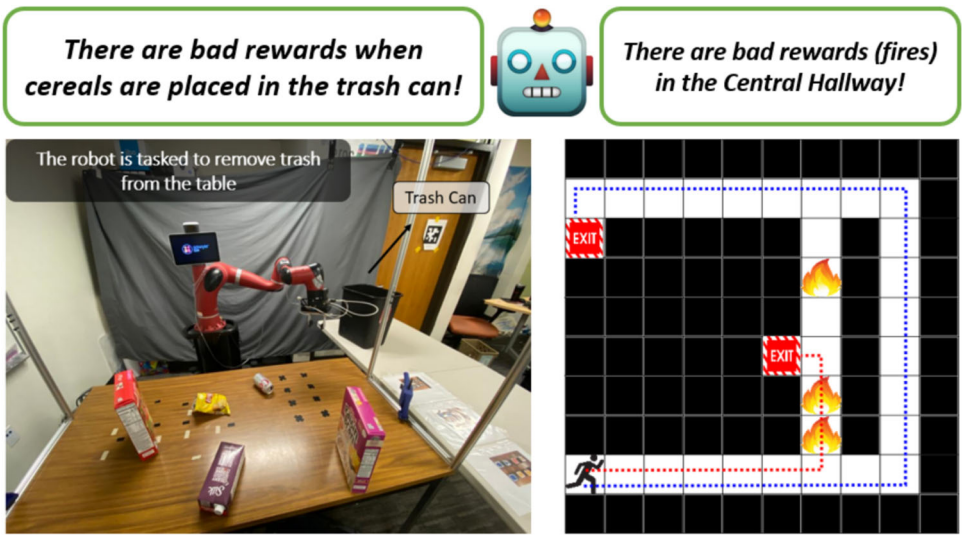

Fig. 1 (Left) The robot is cleaning a tabletop, but the human observer mistakenly thinks all objects will be thrown out, requiring clarification. (Right) A human tries to exit a building during a fire, unaware of hazard locations. With SPEAR, these policies can be repaired (or justified), using natural language updates to produce more optimal behavior

2 Background and related work

Human-Centered Explainable AI. As autonomous systems become increasingly capable decision makers, xAI has emerged as a necessary component for fielding safe autonomous systems. Explainable AI can help bridge the gap between human and autonomous agents by making complex models more understandable, allowing for faster debugging and failure recovery, ultimately improving transparency, trust, and team performance [14–16]. Research in xAI has primarily targeted algorithm transparency for developers, aiding in model debugging and behavior prediction [15, 17]. Though these approaches are beneficial for experts, such methods might restrict end users who engage with these models and directly experience the consequences of failure [18, 19].

A popular approach is to use post hoc explanation methods on RL and/or neural network controllers to enhance interpretability, enabling both developers and novices to understand and debug models during system failures, as well as assist in decision-making [6, 20]. Some examples include generating flowcharts or recipe-like instructions for users [21], employing decision tree-based RL models [22, 23], providing case-based explanations for visualizing class boundaries [24], and using counterfactual explanations to infer a causal link between input and output models [25].

Others have explored the generation of different types of explanations based on user preference [26, 27]. Work by Briggs and Scheutz explored adverbial cues informed by Grice's maxims of effective conversational communication (quality, quantity, and relation) [28] to transparently track and update mental models of collaborators [29]. Others have leveraged abstraction in explanations [6], allowing for simplified and more useful explanations when an agent's decision-making model is too complex for the observer to comprehend. Our proposed approach generates explanations using overstatement or understatement to abstract detail when necessary, enabling agents to provide helpful feedback even with limited common language.

Explanations for Model Reconciliation. Previous research has demonstrated that explanations bring transparency and also play a functional role in synchronizing expectations during misalignments between human and robot agents [6, 7, 30]. Moreover, people tend to trust autonomous agents more when they have a clear understanding of the robot's capabilities and decision-making process [16]. An effective approach for establishing shared mental models in human–robot collaboration has been to use natural language to explain robots' behavior or underlying logic [4, 13]. Hayes and Shah [13] approached the problem of state region description as a set cover problem, trying to find the smallest logical expression of predicates that succinctly describe a target state region. While original, their method's exponential memory and runtime relative to domain and predicate size limits its real-world applicability. This method was further adapted to multiagent RL environments for policy summarization and query-based explanations [31]. Our approach builds on these formulations and enables real-time applicability in most real-world problems.

Furthermore, though these techniques focus on improving an agent's transparency and behavior using explanation, they don't account for collaborators' level of knowledge or need for the information. Recent research has leveraged value of information (VOI) to determine when and what information to communicate during collaborative decision-making [32, 33]. Luebbers et al.'s framework, grounded in VOI theory, allows robots to strategically time justifications during periods of misaligned expectations for greater effect. This approach improves performance, assists users in making informed decisions, and promotes higher interpretability [34].

In this work we focus on *policy elicitation*, a process through which feedback is crafted and given to another agent, in the form of reward function updates during task execution, such that they change their behavior to match the desired policy. By providing reward information for targeted regions of state space through explanations (symbolic updates), we can modify a collaborator's reward function using semantic descriptions.

3 Policy elicitation via social manipulation

The goal of policy elicitation is to cause a behavior change (policy update) in another agent through some form of communicative act. To effectively collaborate with others and coach them toward more optimal policies, it is essential that these communications are intelligible [4, 35, 36], but directly communicating states (i.e. the feature vector itself) relies on there being an efficient mechanism to communicate that information quickly. These criteria are unlikely to hold with humans or heterogeneous robotic agents (not all agents will have the same state representations) as the intended recipients. Our work generates state space-agnostic natural language descriptions of state regions and corresponding reward information (i.e., abstracting the low-level state space), allowing agents to update their reward functions (and policies). The process of eliciting these updates is formally defined in Sect. 4.2, where we present the mathematical structure and objectives of policy elicitation.

Planning Predicates. We define a **base predicate** to be a pre-defined Boolean state classifier (as found in traditional STRIPS planning [37]) with associated string explanation (e.g., in_central_hallway(x) → "X is in the central hallway."). To represent intersections of predicates (e.g., "in the central hallway" AND "has fire extinguisher"), we introduce **composite predicates**, which consist of multiple base predicates and evaluate to true if and only if all base predicate members evaluate to true. Predicates may also have a cost associated with them (e.g., how long they take to communicate) to assist the optimizer with generating more desirable solutions. In SPEAR, predicates are composed in disjunctive normal form to communicate state regions needing reward updates to improve the collaborator's task performance (Fig. 3).

3.1 Formal definition of policy elicitation

Policy elicitation is the process of guiding an agent to update its initial policy, π_{init}, representing its current behavior, to align with a target policy, π_{target}, representing its desired behavior. This is achieved by providing corrective feedback U, which modifies the agent's understanding of its reward structure.

The corrective feedback $U : S \rightarrow \mathbb{R}$ adjusts the agent's initial reward function $R_{\text{init}} : S \rightarrow \mathbb{R}$ by incorporating new information (e.g., explanations, symbolic updates). The updated reward function is defined as:

$$R_{\text{updated}}(s) = R_{\text{init}}(s) + U(s),$$

where $s \in S$ represents a state and $U(s)$ quantifies the adjustment applied to $R_{\text{init}}(s)$.

The agent's updated policy, π_{updated}, is obtained by optimizing for the modified reward function:

$$\pi_{\text{updated}} = \arg\max_{\pi} \mathbb{E}_{\pi}[R_{\text{updated}}],$$

where $\mathbb{E}_{\pi}[R_{\text{updated}}]$ is the expected cumulative reward under the updated policy.

The goal of policy elicitation is to ensure that the updated policy π_{updated} approximates the target policy π_{target} by achieving performance within an acceptable threshold τ:

$$\mathbb{E}_{\pi_{\text{updated}}}(R_{\text{target}}) \geq \tau,$$

where τ is a domain-specific threshold set by a domain expert based on acceptable task performance and R_{target} represents the reward function associated with the target policy.

Assumptions. We characterize the problem of repairing or otherwise manipulating an agent's policy as one of identifying and reconciling divergence between a source and target reward function. Our proposed algorithm relies upon the following assumptions: 1) agents can understand the natural language description of the predicates (mapping string to predicate), and these predicates are grounded in a shared and interpretable state representation

[13]; 2) agents are operating using a functional planner or policy learning algorithm informed by a reward function (or something analogous), and this planner is capable of finding an improved policy given updated rewards [4]; 3) agents' suboptimal action selections are attributable to a malformed reward function rather than a malformed policy search algorithm or inaccurate dynamics model of the environment [4, 38]; 4) the environment transition dynamics remain stationary during policy elicitation, meaning that updated rewards will lead to consistent behavior improvements.

4 Approach

In this section, we specify the problem of reparative policy elicitation: repairing or manipulating an agent's policy by communicating corrections for model divergences. Our approach to policy elicitation uses three components that: 1) estimate the agent's reward function; 2) determine important reward function disparities that prevent desirable behavior; and 3) determine which states to provide reward updates for and communicate a corrective explanation. For clarity, we will use the terminology of an 'expert' to refer to an agent who is initiating communicative action, and a 'novice' to refer to an agent or human whose policy is being corrected.

4.1 Belief estimation with active observation

We formulate our domain as a Markov Decision Process (MDP)[39], wherein an agent acts to maximize an expected reward. M is a MDP defined by the 4-tuple (S, A, T, R) where S is the set of states in the MDP, A is the set available actions, T is a stochastic transition function describing the model's action-based state transition dynamics, and R is the reward function $R : S \times A \times S \rightarrow \mathbb{R}$. Intuitively, M serves as a simulator for an agent in the task domain.

The novice human collaborator's internal policy is not directly observable by the expert agent, making it latent from the expert's perspective. To address this, we perform belief estimation to infer the human's most likely reward function, R_h, based on the observable information, similar to [4, 34]. From this inferred reward function, we derive the corresponding policy, π_h^*, under the assumption that humans act to optimize their expected reward given their current understanding of rewards. This assumption aligns with established practices in inverse reinforcement learning and preference learning literature [4, 38, 40]. Because the only reward information humans receive is communicated either via the expert agent or through observing human behavior, we update the human's reward function R_h and resultant policy π_h^* whenever the agent provides a communicative update or based on human's past actions, similar to [34].

4.2 Finding important model divergences

Once we have our belief of the novice human agent's possible reward function R_h, we identify divergences between the optimal policy π^* and the policy of the human π_h^* that would cause a reduction in the human's expected cumulative reward. We do this by comparing policies trained on R and R_h, where R is the true reward function of the domain and R_h is the reward function used by the novice human. We then identify a set of states $\bar{S}$ for which we communicate updated reward information to augment R_h, such that a more optimal policy (closer to the expected reward of π^*) is elicited (Fig. 3b).

4.3 Communicating state regions

We approach the problem of efficiently describing state regions as a set cover problem, trying to find the smallest logical expression of communicable predicates to succinctly describe target states as in Hayes and Shah [13].

Unlike prior work, we solve for the minimum set cover of the targeted state region using an integer program formulation that admits approximate solutions as well. The inputs to our IP formulation include:

- A set of state indices $\bar{S} = \{s_1, s_2, \cdots, s_{|\bar{S}|}\}$ that correspond to states with expected reward function divergence that need to be communicated for policy repair.
- A set of communicable predicates $\bar{P} = \{p_1, p_2, \cdots p_{|\bar{P}|}\}$. The following is necessary for a solution to exist: $\exists\ \bar{Q} \subseteq \bar{P}$, such that $\forall s \in \bar{S}$, s is covered by a predicate in $\bar{Q}$, — for every state that needs to be covered ($\bar{S}$), there exists a non-empty subset of predicates $\bar{Q}$ that can cover it.
- A set of costs $\bar{C} = \{c_1, c_2, \cdots, c_{|\bar{P}|}\}$, such that $\forall\ c \in \bar{C}, c \neq 0$ — every predicate has nonzero cost associated with using it to update the human's reward function R_h.
- The desired trajectory for the human $\bar{O} = \{o_1, o_2, \cdots, o_{|\bar{O}|}\}$, where o_i describes the state achieved after taking the i^{th} action following the optimal policy π^* from the start state.

The cost for each predicate can be customized per task, as many factors may influence the cost of a predicate. One such criteria for defining cost can be the length of the string describing the predicate. Such a criteria could generate more easily understood explanations by imposing penalties for being too verbose.

A solution to the policy elicitation problem consists of selecting predicates to communicate reward information about specific state regions such that a more optimal policy is produced within some ϵ bound of the optimal policy's expected reward, $|\mathbb{E}_{\pi^*}(R) - \mathbb{E}_{\pi_h^*}(R_h)| \leq \epsilon$.

To minimize this objective while satisfying all the constraints, we define the mathematical formulation of our IP, which we refer to as **SPEAR-IP**, below:

The decision variable x_j is defined as:

$$x_j \in 0, 1, \text{ where } x_j = \begin{cases} 1, & \text{if predicate } p_j \text{ is included in the set cover,} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

using the above definitions of policy elicitation, the objective function is:

$$min \sum_{j=1}^{|\bar{P}|} c_j x_j + L \sum_{k=1}^{|\bar{O}|} \sum_{j=1}^{|\bar{P}|} v_{kj} x_j \tag{2}$$

while subject to the following constraints:

$$\sum_{j=1}^{|\bar{P}|} u_{ij} x_j \geq 1 \quad \forall\ i \in [1, |\bar{S}|] \tag{3}$$

where we define the following matrices:

1. U: $|\bar{S}| \times |\bar{P}|$ matrix representing state-to-predicate coverage, defined as:

$$u_{ij} = \begin{cases} 1, & \text{if state } s_i \text{ is covered by } p_j, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

2. V: $|\bar{O}| \times |\bar{P}|$ matrix representing trajectory-to-predicate overlap, defined as:

$$v_{kj} = \begin{cases} 1, & \text{if trajectory state } o_k \text{ is covered by } p_j, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

L is a large constant that acts as a penalty term and soft constraint violation indicator. The indices i and k are in sets $\bar{P}$ and $\bar{O}$, respectively, and $x_j \in \{0, 1\}$ indicates whether the predicate p_j is included ($x_j = 1$) or excluded ($x_j = 0$) from the set cover. Equation 2 minimizes the cost of the set cover while prioritizing completeness. The first term of Equation 2 minimizes the total cost of the chosen predicates, while the second term penalizes the objective function for any overlap of the selected set cover with the desired path when communicating a negative reward (this can be inverted for positive reinforcement). Equation 3 provides a hard constraint for the inclusion of states from $\bar{S}$ in the set cover. Equation 4 defines the elements of matrix U, which encapsulates cover constraints from $\bar{S}$ (inclusion of all states). Equation 5 defines the elements of matrix V using the desired trajectory $\bar{O}$, encapsulating the requirements for eliciting the desired policy.

The second term from the objective of Equation 2 can be removed to enforce a hard constraint to find an exact set cover that excludes states on the desired path. However, this approach restricts the capability to find solutions that would provide a *near* optimal policy update. This second term leads to **three possible cases**: 1) Set cover solution with a low cost (cost $< L$); 2) No solution for the set cover; and 3) Set cover solution with high cost (cost $> L$).

Cases 1 and 2 are straightforward and describe the ability of SPEAR-IP to solve the set cover. Case 3 is interesting and provides the option of further exploration to find an alternate solution. Using the set cover from Case 3, we can identify overlapping states with the desired path. By penalizing these states in the true reward function R, we can simulate an alternate desired policy, effectively coming up with a contingency for imprecise language. This process of penalization is repeated until either Case 1 or 2 is reached.

SPEAR-IP is a specialized variant of the NP-hard set cover problem, with added constraints and objectives. The runtime varies based on problem characteristics and solver efficiency [41]. We discuss the runtime in Sect. 5.2.

Algorithm 1 Single-shot Policy Elicitation for Augmenting Rewards (SPEAR)

Input: MDP (S, A, T, R), Min. Reward Threshold R_L, Agent Reward Function R_h, Current state s_c, Num. Rollouts k
Output: Semantic Reward Correction

$\bar{S} \leftarrow \emptyset$; $L \leftarrow$large scalar value;
$R_h^* \leftarrow R_h$; // Best possible agent reward function
for *rollout in range(1 to k)* **do**
  $r_c \leftarrow 0$; // Cumulative reward
  $\pi_h \leftarrow$policy trained on R_h;
  // Find states to update for best possible R_h^*
  $s \leftarrow s_c$;
  while *s is not a terminal state* **do**
    // Perform forward rollout of π_h
    $s' \leftarrow T(s, \pi_h(s))$;
    $r_c \leftarrow r_c + R(s, \pi_h(s), s')$;
    $s \leftarrow s'$;
    if $r_c \leq R_L$ **then** // Reward too low
      $\bar{S} = \bar{S} \cup s'$; // Track state for later
      $R_h^*(s, \pi_h(s), s') \leftarrow R(s, \pi_h(s), s')$;
      break;
$\pi_h^* \leftarrow$policy trained on R_h^*; $\pi^* \leftarrow$policy trained on R;
Set_Cover, Objective $\leftarrow$ predicate_selection$(\bar{S}, \pi_h^*, ...)$;
if *objective is no_solution* **then** exit;
if *objective* $\geq L$ *(from Eq. 2)* **then**
  for *rollout in range(1 to k)* **do**
    $s \leftarrow s_c$;
    while *s is not terminal* **do**
      // Perform forward rollout of π^*
      $s' \leftarrow T(s, \pi^*(s))$;
      if $s' \in$ *Set_Cover* **then** $R(s, \pi^*(s), s') \leftarrow -L$;
      $s \leftarrow s'$;
  go to 1;
else
  feedback $\leftarrow$ "There's a bad reward in *Set_Cover*."
  return *feedback*

Algorithm 2 Predicate selection (Minimal set cover)

Input: Set of States to cover $\bar{S}$, Agent policy π, MDP (S, A, T), Set of predicates $\bar{P}$, Current state s_c
Output: Set_Cover and Objective (high, low, or no solution)

Set_Cover $\leftarrow \emptyset$; // Predicates in min set cover
$O \leftarrow \emptyset$; // States we don't want to cover
$s \leftarrow s_c$; $O \leftarrow O \cup s$;
while *s is not terminal* **do**
  // Perform 'optimistic' forward rollout of π
  $s \leftarrow$ most likely transition from $T(s, \pi(s))$;
  $O \leftarrow O \cup s$; //append occupied states
//Define matrix U to be $|\bar{S}| \times |\bar{P}|$ matrix s.t.
for $i \in [1, |\bar{S}|]$, $j \in [1, |\bar{P}|]$

$$u_{ij} = \begin{cases} 1, & \text{if } s_i \in \bar{S} \text{ is covered by } p_j \in \bar{P} \\ 0, & \text{otherwise} \end{cases}$$

//Define matrix V to be $|\bar{O}| \times |\bar{P}|$ matrix s.t.
for $k \in [1, |\bar{O}|]$, $j \in [1, |\bar{P}|]$

$$v_{kj} = \begin{cases} 1, & \text{if } o_k \in \bar{O} \text{ is covered by } p_j \in \bar{P} \\ 0, & \text{otherwise} \end{cases}$$

//For SPEAR-IP: refer to Equations 2 - 3
Set_Cover, Objective $\leftarrow$ SPEAR-IP(U, V);
return *Set_Cover, Objective*

4.4 Algorithm

Algorithm 1 details how SPEAR identifies communicative predicates. We build intuition for this using our emergency evacuation example from Fig. 1-right. Given an estimate of the novice human agent's reward function, SPEAR communicates a reward update in an attempt to elicit the best possible policy. To achieve this, (Line 3–16) we perform multiple forward rollouts of the novice agent's policy π_h derived from R_h and (Line 13–16) compare the accumulated expected reward to the reward threshold R_L. The moment this threshold is crossed, the reward from that transition is determined to be relevant for updating R_h and the state is added to the set cover. The threshold R_L is domain-specific and depends on the threshold for "failure", which will vary across reward functions and the meaning of reward. Note that suboptimal does not necessarily imply failure: failure is a subjective distinction that the domain designer must make.

This can be easily illustrated for our example, where Fig. 2b shows the belief of a novice human trying to evacuate the building. Figure 3b gives insight into how updating the human's reward function can result in an optimal policy even if the human's belief about the fires doesn't truly match the environment. Likewise, for a different building layout, a policy causing a human to take a longer path than optimal may still be considered both successful and an acceptable improvement if it causes the human to safely reach an exit.

After finding the states responsible for meaningful reward divergence, we find a minimal set cover of communicable predicates that map onto these states. This is achieved through **SPEAR-IP**, discussed in Sect. 4.3. We use an off-the-shelf optimizer [42] to solve SPEAR-IP (Line 18), where the *predicate_selection* method takes in the states to cover ($\bar{S}$) and the best agent policy π_h^*. This gives a set of communicable predicates and a final cost using Algorithm 2.

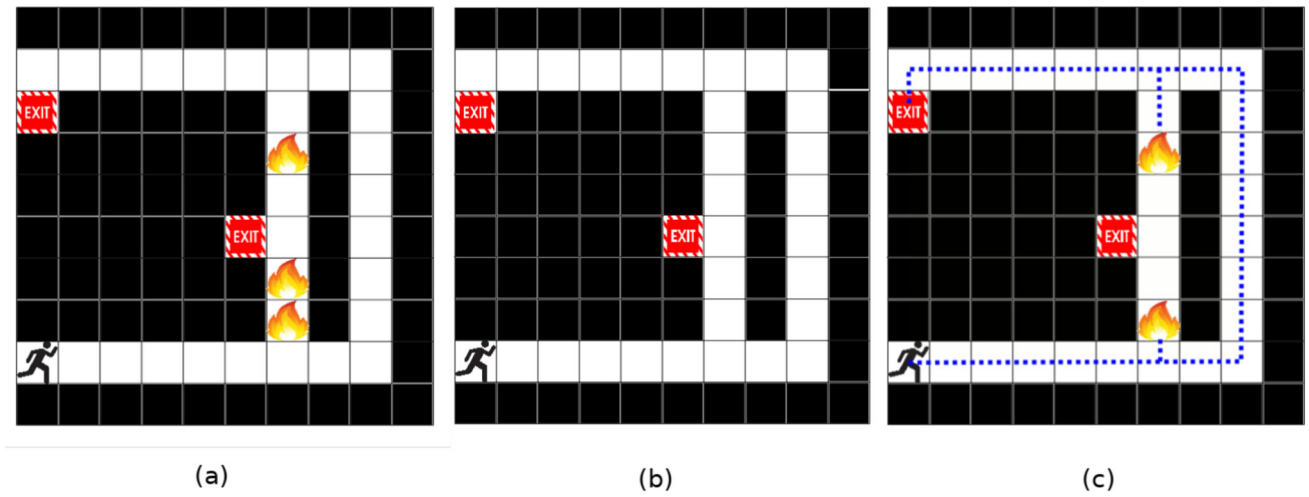

Fig. 2 Rollouts of the human's estimated policy are used to identify suboptimal behavior. **a** The true environment that the agent is acting in. **b** The environment that the human believes it is acting in. **c** A rollout of the human agent's estimated policy reveals the minimal amount of hazardous states that need to be communicated for policy repair

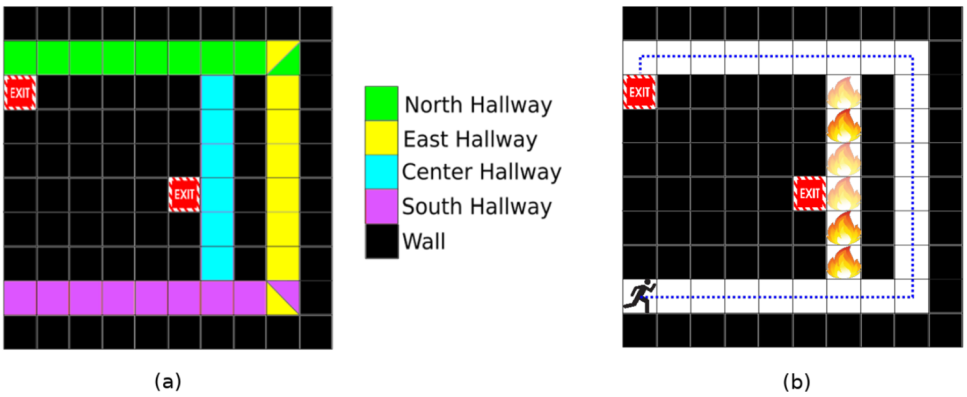

Fig. 3 We ground reward function updates in language using a Boolean algebra over predicates. **a** A domain map with an overlay indicating states where various predicates are true. **b** After communicating "The center hallway is on fire," a belief update for the human shows potential fire locations, though language imprecision means some of the fires (shown as faded) do not exist. Despite the imprecision, the optimal policy is elicited from this update

Line 19 checks whether or not a solution exists for a given set of predicates. In lines 20–28, our algorithm evaluates case 3 to determine if alternate solutions exist. In lines 23–27, SPEAR performs multiple forward rollouts of the optimal policy to find states responsible for a high cost. In line 26, these states are penalized in the true reward function (R) to incentivize the algorithm to find an alternate solution which avoids these states. This enables SPEAR to explore alternate solutions, making it more robust in applications where the available predicates are insufficient, overcoming barriers due to imprecise or unavailable language.

In line 28, now that all the appropriate states are penalized, the algorithm repeats the whole procedure from the beginning with a modified R, continuing this process until it finds a low objective solution or no solution (case 1 or 2). Finally, in line 30, the update is serialized as semantic feedback using the *Set_Cover*. This feedback generation strategy uses negative reward to drive a novice human agent's policy away from undesirable states, as improved policies can then be elicited through the exclusion of states along the human's (hypothesized) originally intended path. While similar outcomes can be achieved via positive reward, a state exclusion-based strategy generally allows for the use of less precise predicates.

In Algorithm 2, we produce the set cover for communicating the reward update. Lines 4–7 evaluate states we want the novice agent to traverse (the desired trajectory $\bar{O}$) by performing a forward rollout of the best attainable policy π_h^*. In lines 8–13, the matrices U and V from Equation 4-5 are defined, which form the basis of the constraints governing the inclusion of states to cover and exclusion of optimal states (when giving information about negative reward) in the set cover, respectively. Finally, in line 15, *SPEAR-IP* is solved using the matrices U and V to give *Set_Cover* and *Objective*.

5 Algorithmic evaluation

To demonstrate the utility of our algorithm and validate the effectiveness of its generated explanations, we present a series of algorithmic evaluations and human-subjects user studies. This section focuses on an empirical analysis of SPEAR's algorithmic performance, considering domain size and predicate count in a simulated emergency evacuation scenario. In Sects. 6 and 7, we present two separate human-subjects studies aimed at evaluating the usefulness and applicability of explanations generated by SPEAR across various applications, targeting expert and novice users, respectively.

5.1 Evaluation domain

In this scenario, our expert autonomous agent must help a human agent escape from a smart building in which a series of fires has broken out, as shown in Fig. 2. While people in the building are not aware of the fire locations, they are assumed to understand the generated predicate-based language.

The building layout for each trial is generated with a stochastic placement of rooms, hallways and exits over a gridworld of fixed size. For example, a gridworld of size 40 × 40 (1600 states) may have parameters: rooms = 10, hallways = 40, and number of exits = 5. We utilized randomly generated predicates for validation to demonstrate the generalizability of our approach. To accommodate the full range of possible state regions to cover using predicates, we consider composite predicates (intersections of base predicates) from the power set of base predicates, providing each scenario with an upper bound of $2^n - 1$ possible predicates given n base predicates.

Once the building layout and predicates are generated, we observe a randomly placed human agent exploring a hazard-free building over a fixed number of episodes (e.g., 25 episodes for 40 × 40 states). The human agent's policy is then trained to always seek the shortest path to the closest exit that was discovered during these exploration episodes. Initially, SPEAR has no knowledge about which exits the human is aware of, but gradually, its belief about the human's reward function is updated from these past observations [34]. Next, we begin the evaluation by adding fire to the building randomly. Once this process completes, we start the SPEAR evaluation. Predicates from SPEAR-IP are used to update the human agent's reward function during the episode. We update the policy of the human using the repaired reward function after each update.

5.2 Algorithmic performance

We evaluate performance based on state space size and predicate count using various building layouts. We test its adaptability to diverse state mappings with randomly generated, n-sphere shaped predicates. In structured environments like Fig. 3a, building-grounded predicates make it easier to find hazardous predicates. However, stochastic predicate placements, which simulate non-tailored languages, complicate solutions and increase computational time, highlighting the algorithm's resilience in predicate-domain mismatches.

We assess our algorithm in an evacuation scenario for both deterministic and stochastic environments (Fig. 4). In the stochastic domain, a stochastic transition function was applied describing the model's action-based state transition dynamics (agent takes prescribed action with a probability of 85%). For each map in both environments,

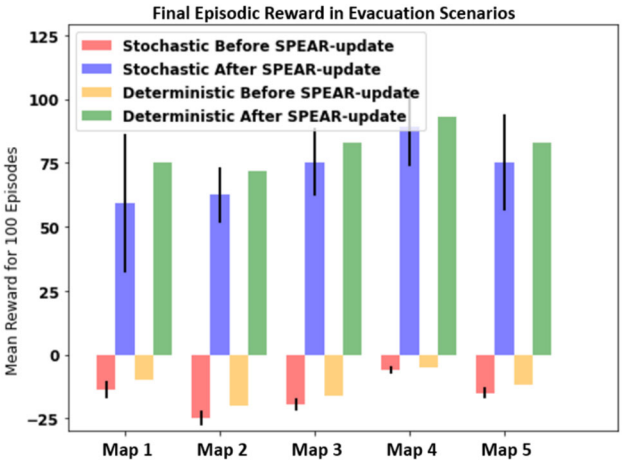

Fig. 4 SPEAR's evaluation in stochastic and deterministic evacuation domains (25×25) shows substantial increase in episodic reward from symbolic reward updates

we evaluate our algorithm for 100 episodes. In each episode the reward was computed both before and after the SPEAR update (exit: $+100$, fire: -100, and each step: -1). For our stochastic evaluation, we set our forward rollout count parameter, k, to 10. The results from our simulations show a substantial improvement in the episodic reward after the SPEAR update (Fig. 4), demonstrating utility in both deterministic and stochastic domains.

Furthermore, we analyze performance as a function of predicate count by dividing into the two success cases described in Sect. 4.3: 1) Maps with a low cost solution (Case 1); and 2) Maps with high cost solution (Case 3). Figure 5-left shows how algorithm performance changes with increasing predicate count on low cost maps. For Case 1 solutions, our algorithm exhibits linear computational time as a function of the number of predicates, with the set cover able to be computed within 50 s with nearly 10,000 communicable predicates. To put the significance of this in context, the prior state-of-the-art using the Quine-McCluskey(QM) algorithm [13] takes approximately 60–120 s for solving set covers with 10 predicates on similar hardware, with performance deteriorating exponentially as the predicate count increases. Our approach achieves an order-of-magnitude improvement over the QM-based method, operationalizing insights from past work [13] to communicate state regions—underpinning SPEAR's policy update.

We plot the performance for Case 3 solutions as predicate count increases (Fig. 5-right). We observe that the plots again scale linearly in practice with respect to the number of predicates, but with higher computation time

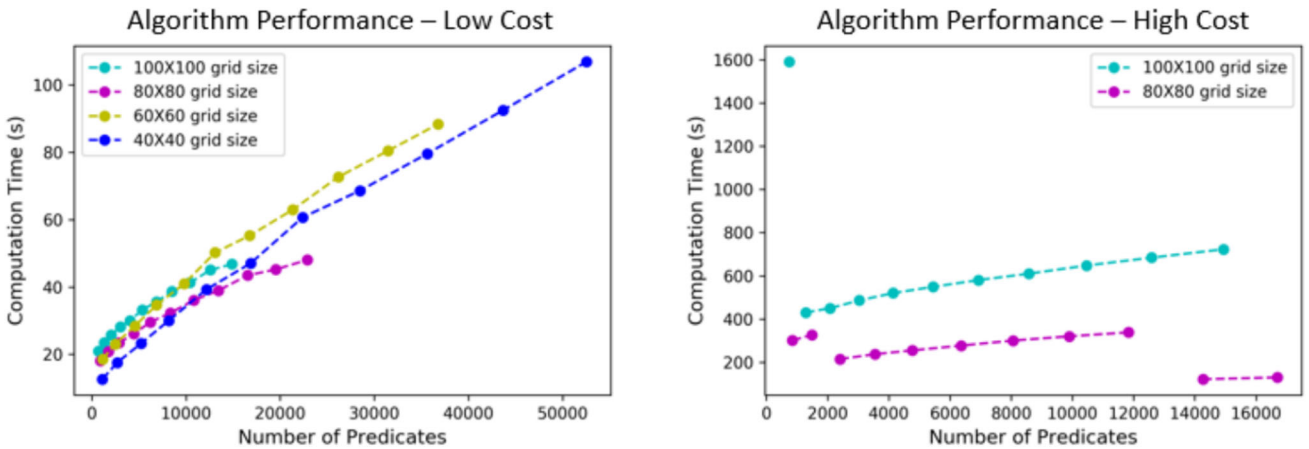

Fig. 5 (Left) SPEAR's performance on low-cost (case 1) maps shows linear runtime growth with predicate count. (Right) High-cost map (case 3) performance scales linearly with predicate count; sharp time drops (purple) result from SPEAR more quickly finding solutions with new predicates

due to multiple SPEAR runs. Computation time is higher here because the algorithm has to explore alternative solutions for the desired policy, solving for set cover solutions multiple times (Sect. 4.3).

SPEAR achieves a dramatic improvement in computation time over the QM-based set cover approach by pre-computing predicates over the state space, which alleviates some of the online computation during the prime implicant step (exponential in computational time) of the QM approach [43]. Additionally, QM can only provide an exact set cover, failing when there is no exact cover, making it slower and less versatile. SPEAR's use of an approximation heuristic makes it faster and more adaptable than the QM-based approach.

We observed interesting irregularities in performance as evident by results from the 80 X 80 world (Fig. 5-right-purple). We find that sudden dips in computation time can occur when new language enables the algorithm to find a Case 1 (single-loop) or cheaper Case 3 (multiple loops, but fewer) solution.

The final part of our performance analysis looks algorithm performance as domain size increases with fixed predicate count (100 base predicates). We generate a set of maps with similar parameters for a fixed state space size, sampling from these maps to get the mean and confidence bound for 10 simulation runs as shown in Fig. 6. A significant take-away from this analysis is the insight that an attention mechanism is more important for abstracting and reducing the domain's state space than for limiting the number of predicates to consider, as prior work anticipated [4, 13, 36].

Result Synopsis We have shown that *SPEAR* enables substantial improvements in agent performance through policy elicitation, through a novel method for communicating about state regions that substantially outperforms prior work. These contributions enable semantically guided policy manipulation for a much broader class of problems than was previously possible, providing a method that scales linearly with predicate count as opposed to exponentially, advancing the state-of-the-art in autonomous coaching through new algorithms and improved foundational capability.

6 Study 1: explanation quality evaluation

We evaluated SPEAR explanations through an IRB-approved online user study (n = 12) from a population of graduate researchers in AI and robotics unaffiliated with this line of work in order to solicit feedback from participants with domain familiarity. The study evaluated the effectiveness of these explanations, analyzed the benefits of varying levels of abstraction in explanations, and determined their impact on user comprehension.

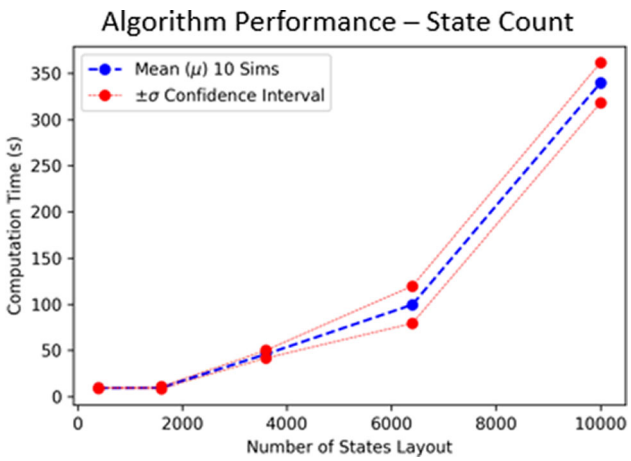

Fig. 6 SPEAR's evaluation in stochastic and deterministic evacuation domains (25 × 25) and its performance as state space increases, revealing polynomial computation time with state space

6.1 Experimental design

The experiment was structured as a single remote session where participants completed five tasks. For each task, participants were shown an image of a scenario featuring a human agent with specific objectives. They were informed that some scenario details were concealed and would be later revealed by an autonomous agent through updates. Participants were tasked with choosing the optimal action from multiple-choice options, similar to [44].

Subsequently, an image illustrating the human agent's optimal behavior was presented, accompanied by three possible updates from the expert autonomous agent. Participants ranked these updates using subjective metrics from established questionnaires [45, 46], considering which update, if received earlier, would most likely lead to the desired behavior. This procedure was repeated for all five tasks. Upon completing the tasks, participants were administered a demographic survey and an open-ended post-experiment survey.

6.2 Explanation updates

Participants were presented with three distinct types of explanation updates:

C1. *No Set Cover:* Here, the agent conveys good or bad states in the task by utilizing machine language in conjunction with a string template (e.g., "There are bad rewards when Obj_1 and Obj_5 are in states: {x =(6.5−7.5), y = (6.25)}."). Essentially, it provides information about crucial states with model divergence without employing any set cover to translate them into natural language updates.

C2. *Exact Set Cover:* This approach is similar to C1, but key states with model divergence are communicated using exact set cover conditions (e.g., "There are bad rewards when the red cereal and purple cereal are placed in the trash can."), similar to [13].

C3. *Relaxed Set Cover:* This approach is similar to C2, but key divergent states are conveyed with a focus on abstraction, although at the cost of specificity (e.g., "There are bad rewards when cereals are placed in the trash can."). Example explanations provided here correspond to the scenario in Fig. 1-left.

Both 'exact set cover' and 'relaxed set cover' explanations were generated utilizing the SPEAR-IP formulation.

6.3 Experimental tasks

We chose three distinct domains for wider applicability and generalizability. Task 2 and Task 4 specifically test both positive and negative reward updates.

Task 1 and Task 2 *Navigation Tasks* An agent traverses a gridworld domain from start to goal using the most cost-effective path. Path costs vary based on good or bad state encounters, with some grid cells colored to facilitate task abstraction and to showcase language grounding, inspired by [44].

Task 3 and Task 4 *Fire Rescue Tasks* Here, an agent attempts to rescue victims from a burning building, searching a limited number of floors sequentially. The user is not aware of which floors or rooms have fire or victims, which is later communicated via updates.

Task 5 *Robotic Cleaning Task* A robot removes trash from a tabletop, leaving some items believed to be trash by participants. In this scenario, the expert agent justifies its actions by providing explanations to the participants (Fig. 1-left), similar to [4].

6.4 Hypotheses

We hypothesized that users would rate the 'relaxed set cover' explanations higher than both the 'no set cover' and 'exact set cover' explanations **(H1)**, based on the following metrics: usefulness (H1a), conciseness (H1b), comprehension (H1c), cognitive load (H1d), decision-making (H1e), and interpretability (H1f). We also

hypothesized **(H2)** that participants would prefer structured semantic explanations over numerical reward explanations (i.e., 'exact set cover' > 'no set cover') across the same subjective metrics as H1 (H2a-H2f).

6.5 Results and analysis

We recruited 12 participants (6 males and 6 females) with ages ranging from 23 to 40 years old ($M = 27.92$; $SD = 4.66$). This sample size would allow us to detect an effect size of $f = 0.50$ with .95 power at an alpha level of .05 (calculated using the software G*Power [47]). For each task, we assessed the explanation updates on ranked data provided by the participants using a nonparametric Kruskal–Wallis test with explanation updates as a fixed effect. Post hoc comparisons used Dunn's test for analyzing explanation types for stochastic dominance.

Task 1 and Task 2. We found a significant effect in favor of the 'relaxed set cover' update over 'no set cover' and 'exact set cover' for usefulness, conciseness, comprehension, cognitive load, decision-making, and interpretability, with all p-values below the 0.05 threshold for both Task 1 and Task 2. For Task 1, post hoc analysis with Dunn's Test indicated that participants consistently preferred 'relaxed set cover' over 'no set cover' for usefulness, comprehension, cognitive load, and interpretability ($p < 0.05$). Similarly, for Task 2, Dunn's Test revealed that participants rated 'relaxed set cover' over 'exact set cover' across all measures with higher means ($p < 0.0001$), **validating H1a-H1f** for both tasks.

Interestingly, in both Task 1 and Task 2, 'no set cover' had higher means over the 'exact set cover'. For Task 1, Dunn's Test also revealed a preference for 'no set cover' over 'exact set cover' in terms of conciseness ($p < 0.01$), thus **invalidating H2b**. Similarly, for Task 2, we found the same significant results across all measures ($p < 0.05$) , thus **invalidating H2a-H2f** for Task 2. We posit that the preference for 'no set cover' over 'exact set cover' may derive from the simplicity of these navigational tasks, implying that in simpler scenarios, providing full reward information in numerical form might be more effective than converting to semantic form. This partially conforms to results from [44].

Task 3 and Task 4. For both tasks, post hoc analysis using Dunn's test favored the 'relaxed set cover' over 'no set cover' across all measures ($p < 0.01$), **validating H1a-H1f**. There were higher mean ratings in Task 3 for 'relaxed set cover' over 'exact set cover' with respect to cognitive load ($p = 0.021$) and interpretability ($p = 0.002$). Post hoc tests on 'exact set cover' and 'no set cover' revealed that participants preferred the former over the latter for all measures except interpretability ($p < 0.05$), **validating H2a-H2e**—the opposite of what we saw in first two tasks. Dunn's test also shows significant differences for Task 4 between mean ratings of 'relaxed set cover' and 'exact set cover' for comprehension ($p = 0.021$) and interpretability ($p = 0.003$). Similar to Task 3, post hoc tests on 'exact set cover' and 'no set cover' show a higher mean of the former over the latter for all the subjective metrics except comprehension ($p < 0.05$), **validating H2(a,b,d,e,f)**. This demonstrates a stronger preference for structured explanation as domain complexity increases.

Task 5. Post hoc analysis using Dunn's test showed that participants favored the 'relaxed set cover' over the other conditions ($p < 0.001$) across all measures, **validating H1a-H1f**. This demonstrates a preference toward structured explanations over numerical, **validating H2a-H2f**.

Post-experimental Survey. Participants were asked which update they would prefer to use if they had to complete more tasks. 'Relaxed set cover' was preferred over alternatives. Based on a one-sample test of proportions, 9/12 participants chose 'relaxed set cover'; a greater proportion than the expected random proportion of 0.33 ($z = 3.09$, $p = 0.002$). Participants justified their choice for 'relaxed set cover' citing reasons such as conciseness, ease of understanding, and reduced cognitive load, which align with our subjective findings from the experiment.

Result Synopsis. In most tasks and measures, 'relaxed set cover' consistently outperformed both 'no set cover' and 'exact set cover'. Additionally, 'no set cover' performed better than 'exact set cover' in the simpler tasks but the trend reversed with the increase in task complexity, *indicating the need for structured and simpler explanations as the task complexity increases.* The degree to which 'relaxed set cover' outperformed the other two varied among tasks and measures. For example, in the robotic cleaning task, 'relaxed set cover' significantly

outperformed the other updates in all measures. This pattern suggests that the advantages of relaxed explanations become more apparent with increases in the state space and task complexity. In summary, *our findings strongly suggest that SPEAR-based reward explanations are not only more useful than numeric ones but also promote better understanding, decrease cognitive load, and improve interpretability.*

7 Study 2: explanation type evaluation

We conducted a follow-up IRB-approved study ($n = 38$) with a user base recruited from the Prolific platform (prolific.co). The focus of this study was to assess the utility of providing state-based reward updates as opposed to traditional plan explanations [8] (i.e., step-by-step descriptions of a plan) in terms of task performance and task awareness.

7.1 Experimental design and protocol

The study utilized a 2×1 between-subjects experimental design to evaluate two types of semantic guidance: (1) plan explanation, referred to as the 'prescriptive' condition, and (2) reward explanation, known as the 'descriptive' condition.

The experiment was administered online in several batches with randomly assigned conditions using the Prolific platform. To ensure higher participant quality, we filtered for those who had both completed at least 100 approved studies on Prolific and had an approval rate of 95% or higher. This study's methodology mirrors the approach taken in Study 1, mentioned in Sect. 6 where participants completed the same five tasks.

For each task, participants were presented with the same scenarios as in the previous study. They were informed that some details of the scenario were concealed and that an autonomous agent would assist them in solving the task. We initially tested their baseline knowledge and policy preferences by asking them to select the optimal action from multiple-choice options. Following this initial assessment, an update was provided based on the experimental condition (more details in the next subsection). Participants made their updated choices based on the new information. Upon completing all tasks, participants were given a demographic survey, a post-experimental questionnaire, and an open-ended survey.

7.2 Explanation updates

We presented participants with two distinct types of explanation updates depending on their experimental condition:

(1) *Plan Explanation:* Here, the robotic agent communicates a step-by-step plan to a human teammate. For example, *"Begin at your current position. Move one cell to your right...."* These explanations are generated by the GPT-4 model [48], prompted by the free-form text provided by participants in Study 1, where they described the robot's actions step-by-step. These final explanations were checked for correctness by experts.
(2) *Reward Explanation:* These explanations are similar to those presented in the 'relaxed set cover' condition of Study 1 (e.g., *"There are bad rewards in the purple and orange cells."*).

7.3 Experimental tasks

We utilized the same three distinct domains and five tasks from Study 1, comprising two navigational tasks (Tasks 1 and 2), two fire rescue tasks (Tasks 3 and 4), and one robotic cleaning task (Task 5).

7.4 Hypotheses

Through a human-subjects study, we evaluate the following four hypotheses, partitioned into two categories:

H3: Subjective Hypotheses

H3.a: Participants will find the reward explanation more trustworthy than the plan explanation, as the transparency of the recommendation leads to increased trust [4, 49, 50].

H3.b: Participants will perceive the plan explanation condition as less stressful and demanding compared to the reward explanation condition, due to the presence of clear recommendations.

H4: Objective Hypotheses

H4.a: Participants will have a better understanding of the task when provided with a reward explanation compared to a plan explanation, as it offers insight into the task through agent updates.

H4.b: Participants will perform similarly in both conditions (i.e., they will make the correct policy choice after an update is provided).

7.5 Measurement

We recruited 39 participants via the Prolific platform but excluded one due to failing attention checks, leaving 38 participants (21 male, 16 female, 1 unspecified), with 19 participants per condition, aged 19–70 years ($M = 38.16$; SD $= 13.58$). Of these, 16% reported working in STEM fields, and 68% reported having a bachelor's degree or higher. This sample size would allow us to detect an effect size of $f = 0.50$ with .95 power at an alpha level of .05, based on a repeated measures ANOVA between factors design (calculated using G*Power software).

We evaluated our hypotheses using both subjective and objective measures. We administered a post-experimental survey consisting of 7-point Likert scales, similar to the one from Study 1 [51–53]. Based on these questionnaires, we identified two key concepts to validate our hypothesis: *Trust and Workload.*

The *Trust* scale consists of 4 items: confidence, reliability, trust, and dependability (Cronbach's $\alpha = 0.98$). *Mental Load* scale consists of 3 items: demandingness, hurriedness, and effort (Cronbach's $\alpha = 0.82$).

For objective metrics, we recorded the following items: *Policy Accuracy (PA)*, the total number of tasks for which a participant chose the correct policy; and *Knowledge Accuracy (KA)*, the total number of tasks for which a participant had accurate final knowledge.

7.6 Results and analysis

Subjective Analysis. To test our subjective hypotheses, we analyzed post-experiment 7-point Likert scales using independent samples t-tests for between-condition comparisons.

The trust scale revealed a significant effect favoring the 'descriptive' condition over the 'prescriptive' condition $[t(36) = -2.74, p = .009]$, with higher trust scores $(M = 5.83)$ versus $(M = 4.66)$, **validating H3.a**.

The mental load scale results showed no significant difference $[t(36) = -1.83, p = .076]$, with mean scores of $(M = 3.56$ for 'descriptive') and $(M = 2.73$ for 'prescriptive'), respectively. Due to the lack of statistical significance, **H3.b is inconclusive**, and more data are required to definitively address this hypothesis.

However, participants provided insights that pointed toward two interesting trends. First, some participants who were unsure and unconfident, interpreted the robot's direct guidance in the 'prescriptive' condition as a sign of confidence, leading them to stop thinking critically.

- "*I followed the directions from the agent closely and ignored what I first selected.*"

Second, some participants expressed confusion and were seeking more information and understanding when following prescriptive explanations.

- "*They [explanations] were useful but didn't explain why the decision was the best...*"
- "*...explanations were useful, but the system assumed I understood the context of the situation without explaining anything.*"

Whereas for the 'descriptive' condition, many participants not only found the explanations useful but also felt that they encouraged active thinking patterns, leading to a positive impact on trust, as evidenced by the quantitative results:

- "*The information made me make more calculated decisions.*"

These results align with previous findings in the xAI literature, suggesting that some individuals may *over-trust the guidance if they lack confidence in themselves, while others may under-trust and become frustrated if they lack sufficient rationale for the robot's guidance* [49, 54–56]. Furthermore, insights into the robot's decision-making process can encourage people to engage in more active thinking patterns [34, 57, 58].

Objective Analysis. To assess our objective hypotheses, we analyzed task metrics using a nonparametric Mann–Whitney test to evaluate differences between conditions.

First, we conducted a Mann–Whitney test to assess if there were differences in knowledge accuracy (KA) between conditions. The analysis revealed statistical significance in favor of the 'descriptive' condition with $M = 4.11$ compared to the 'prescriptive' condition with $M = 1.82$ for KA scores, with $z = -4.394$ and $p < .0001$, suggesting that participants in the descriptive condition demonstrated higher knowledge accuracy per task scores than those in the prescriptive condition. These findings serve to **validate H4.a**. On the other hand, no statistical differences were observed in policy accuracy (PA) scores between the 'descriptive' with $M = 4.31$ and 'prescriptive' with $M = 4.74$ conditions, with $z = 1.42$ and $p = 0.157$. These results **support H4.b**, indicating that participants in both conditions achieved higher PA scores and predominantly made the correct choice after an update was provided (refer to Fig. 7). Therefore, these findings suggest that while participants in both conditions were able to identify the correct policy, those in the 'descriptive' condition not only identified the correct policy but also exhibited better task awareness.

Result Synopsis In our study, the reward-based explanations consistently outperformed the plan explanations in fostering a higher perception of trust and enhancing users' task understanding, thus improving their policy decisions. Due to the opaqueness of plan explanations, this approach failed to update users' task awareness and led some people to overtrust the guidance. These results suggest that the utility of reward explanations extends beyond mere decision support, offering insights into the robot's decision-making process and fostering a better understanding of tasks, thus promoting active thinking patterns in users.

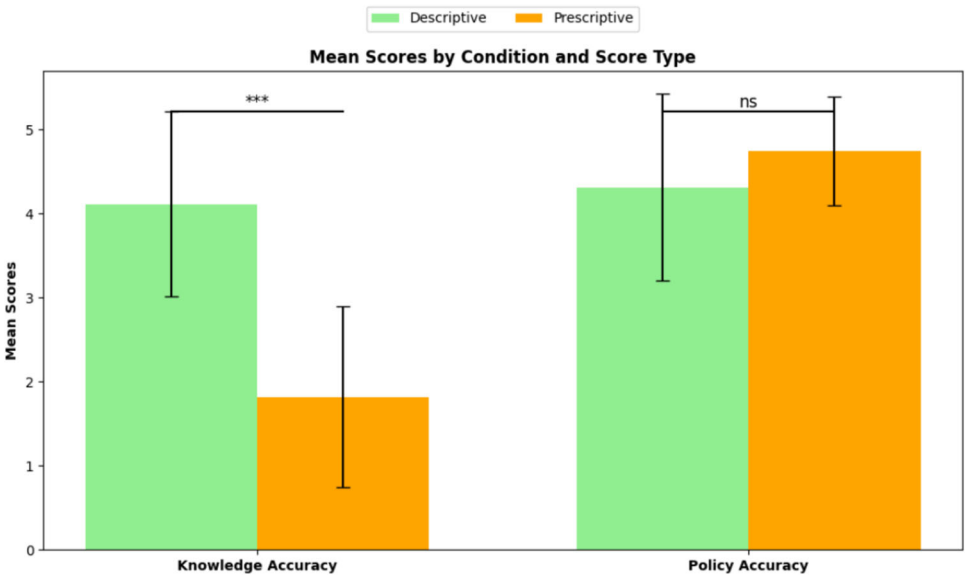

Fig. 7 Comparison of mean scores for knowledge and policy accuracy between descriptive and prescriptive conditions. Results from the Mann–Whitney test highlight that the descriptive condition (reward-based explanations) not only facilitated correct policy elicitation but also enhanced task awareness, outperforming the prescriptive condition (plan explanations)

8 Robotic application: multiagent cleaning

In this section, we illustrate the application of the policy elicitation process in a scenario involving two heterogeneous robotic agents, specifically robots operating with different state representations. This effectively demonstrates the utility of SPEAR's symbolic reward update in facilitating agent-to-agent manipulation.

Our core insight of policy elicitation relies on modifying an agent's behavior to a desired policy by updating their reward function with targeted feedback in the form of semantic explanations (symbolic updates) during task execution (refer Sect. 3 for more details). This allows our method to operate at a level of abstraction that does not depend on the recipient's underlying state space representation, instead only requiring a similar vocabulary of planning predicates.

In many applications, multi-agent robotic systems have no guarantee of operating with identical state representations. Similarly, private entities, such as those in autonomous driving, may not want to share their proprietary reward functions with competitors. Therefore, we leverage common grounding (e.g., language) to communicate updates for efficient behavior modification during the task for repairing another agent's policy.

Assumptions. We implicitly assume a shared predicate grounding between multiple agents (i.e., the same predicate, even if they have different state mappings, generally means the same thing to each agent) even if they operate using different state representations, such that a common symbolic policy update is feasible. While the predicate grounding need not be exactly the same, as different state spaces are unlikely to have one-to-one mappings, we assume the possibility of having a shared natural language via a set of predicates even if it is not exhaustive or comprehensive.

8.1 Robot-to-robot policy elicitation

Here, one agent needs to correct the policy of a second robotic agent to prevent it from removing specific items during a pick and place task (Fig. 8). Importantly, these robots do not operate in the same state space, and therefore cannot directly communicate reward function updates to each other. In this scenario, the Rethink Robotics Sawyer has been tasked with clearing *trash* from a table, and is operating with the malformed belief that all objects on the table's surface are *trash*. The Kinova Movo is observing this scene and has accurate knowledge about the environment (i.e., it knows that certain objects in the scene aren't trash). As Sawyer's reach expresses intent to remove one of the items that isn't trash, Movo detects that Sawyer's policy is incorrect. Movo corrects Sawyer's behavior by providing an update for its reward function, allowing Sawyer to compute a better policy. We accomplish this reward update step by generating semantic expressions about the reward function in parts of the state space (i.e., predicate-grounded natural language communicating about a region of negative reward) for Sawyer using SPEAR.

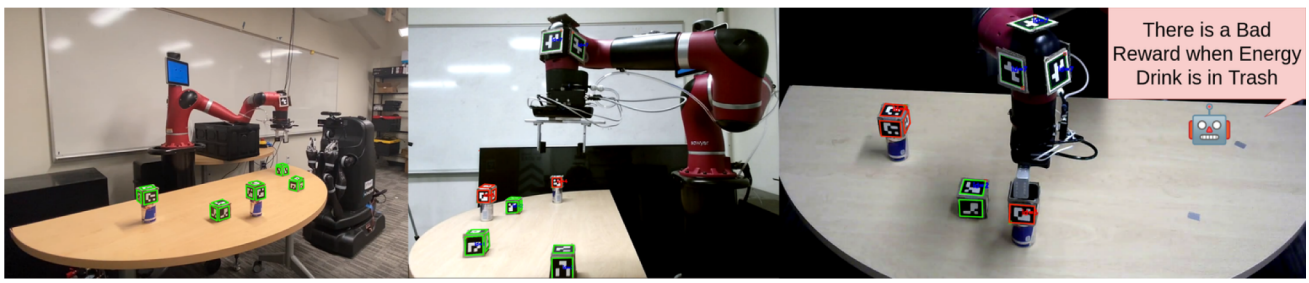

Fig. 8 (Left) Sawyer (red robot) views a misinformed policy for a tabletop cleaning task as Movo (black robot) moves into position to assist. (Center) Movo, from its perspective views the scene with the correct policy and watches Sawyer remove the correct (green) objects from the table. Movo then observes Sawyer reach for one of the incorrect (red) objects and delivers a verbal reward update. (Right) Sawyer computes an update to its planner thus aligning with the desired policy

Within this task, Sawyer used a series of ArUco markers [59] to track the 6-DOF poses of various objects (states) on the tabletop. Using these poses, Sawyer systematically transferred all items classified as *trash* to the waste bin using an interruptable pick and place action. As Movo observes Sawyer, it detects Sawyer's end effector reaching for one of the energy drinks on the table, thus indicating the intent to remove that object, and signaling to Movo that Sawyer's policy was malformed. Using SPEAR, Movo is able infer Sawyer's erroneous belief from this observed action (that a non-negative reward is associated with putting the energy drink in the trash). Movo performs a forward rollout with the inferred policy of Sawyer to predict which state transitions with *negative reward* Sawyer will reach. Next, Movo computes an updated policy for Sawyer by determining which states should be assigned negative reward and which of the available predicates are needed for communicating it with Algorithm 2. A DNF formula of predicates that covers the desired states is then computed and communicated by Movo through natural language: "*There is a Bad Reward when energy drink is in the trash*". Finally, Sawyer uses shared predicates to map the communication into its own state space, update its reward function accordingly, and reconverge a repaired policy, showing successful application of the SPEAR framework.

9 Discussion and conclusion

Limitations and opportunities. Our policy elicitation formulation relies on belief estimation. If the inference of the human's reward function R_h is poor, it will also lead to the degradation of SPEAR's feedback quality for policy elicitation. To accommodate this, we formulated our design to be modular, allowing each component to be easily replaced by any state-of-the-art method [40, 60–62], provided it takes similar input and provides the required output.

Additionally, we assume users can interpret natural language descriptions of predicates, which we validated through our user study. However, predicate interpretability can vary, often depending on the creator and individual scenario, potentially necessitating the hand-engineering of these predicates for each domain. A possible avenue for future research could explore using large language models (LLMs) to make this predicate generation more robust and generalizable. Similarly, in this work, we use string templated responses for policy elicitation, which is not ideal for a conversational agent. Policy explanations from autonomous agents are expected to be conversational and dialogue-based [19, 63]. A direct extension of this work would look into integrating an LLM framework with predicate-based grounding for human reward coaching, leveraging language grounding from predicates alongside the broad contextual generalizability of LLMs.

Finally, our method requires specifying a reward performance threshold τ to determine what constitutes "acceptable" behavior improvement. This threshold is domain-specific and typically tuned by an expert, reflecting what is considered a failure or suboptimal outcome in a given context. By adjusting τ per domain or even per user, one can tailor the stringency of policy corrections to match different risk tolerances and performance expectations, thereby enhancing user experience and accessibility [64, 65]. For example, novice users might prefer a more lenient τ for a gradual learning curve, whereas high-stakes applications might use a stricter τ for safety.

Conclusion. In this work we define the problem of *policy elicitation*, the manipulation of a human's behavior through the use of semantically (natural language) grounded reward updates, and present an optimization-based approach for solving it at scale. We introduce a novel integer programming-based algorithm that renders policy explanation [13] and policy manipulation [4] techniques feasible for use in applications substantially larger than previously possible. Our method leverages relaxed explanations, using overstatement or understatement, to deliver concise and useful feedback even with limited shared language. We demonstrate the utility of these policy explanations for both expert and novice users through a series of human-subject studies. Our results indicate that these relaxed reward-based explanations not only enhance individuals' policies but also decrease cognitive load and improve decision-making, all while preserving interpretability. Additionally, we show that these explanations provide insights into the robot recommender's decision-making process, foster a better understanding of tasks, and thus promote active thinking patterns in users, while also facilitating the desired correction of policies.

Acknowledgements This work was funded by the Army Research Lab STRONG Program (#W911NF-20-2-0083)

Data availability The datasets generated during or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Belpaeme T, Kennedy J, Ramachandran A, Scassellati B, Tanaka F (2018) Social robots for education: a review. Sci Robot 3(21):5954
2. Leite I, Martinho C, Paiva A (2013) Social robots for long-term interaction: a survey. Int J Soc Robot 5(2):291–308
3. Leyzberg D, Spaulding S, Scassellati B (2014) Personalizing robot tutors to individuals' learning differences. In: Proceedings of the 2014 ACM/IEEE International Conference on Human-robot Interaction, pp 423–430. ACM
4. Tabrez A, Agrawal S, Hayes B (2019) Explanation-based reward coaching to improve human performance via reinforcement learning. In: 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp 249–257. IEEE
5. MacLellan CJ, Koedinger KR, Matsuda N (2014) Authoring tutors with simstudent: an evaluation of efficiency and model quality. In: Intelligent Tutoring Systems: 12th International Conference, ITS 2014, Honolulu, HI, USA, June 5–9, 2014. Proceedings 12, pp 551–560. Springer
6. Chakraborti T, Sreedharan S, Kambhampati S (2020) The emerging landscape of explainable automated planning & decision making. In: IJCAI, pp 4803–4811
7. Tabrez A, Luebbers MB, Hayes B (2020) A survey of mental modeling techniques in human–robot teaming. Curr Robot Rep, 1–9
8. Chakraborti T, Sreedharan S, Grover S, Kambhampati S (2019) Plan explanations as model reconciliation. In: 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp 258–266 . IEEE
9. Valmeekam K, Marquez M, Sreedharan S, Kambhampati S (2023) On the planning abilities of large language models–a critical investigation. arXiv preprint arXiv:2305.15771
10. Tamkin A, Brundage M, Clark J, Ganguli D (2021) Understanding the capabilities, limitations, and societal impact of large language models. arXiv preprint arXiv:2102.02503
11. Robinette P, Li W, Allen R, Howard AM, Wagner AR (2016) Overtrust of robots in emergency evacuation scenarios. In: The Eleventh ACM/IEEE International Conference on Human Robot Interaction, pp 101–108. IEEE Press
12. Lage I, Chen E, He J, Narayanan M, Kim B, Gershman S, Doshi-Velez F (2019) An evaluation of the human-interpretability of explanation. arXiv preprint arXiv:1902.00006
13. Hayes B, Shah J (2017) Improving robot controller transparency through autonomous policy explanation. In: Proceedings of the 12th ACM/IEEE International Conference on Human-Robot Interaction
14. Arnold M, Bellamy RK, Hind M, Houde S, Mehta S, Mojsilović A, Nair R, Ramamurthy KN, Olteanu A, Piorkowski D et al (2019) Factsheets: increasing trust in ai services through supplier's declarations of conformity. IBM J Res Dev 63(4/5):1–6
15. Ribeiro MT, Singh S, Guestrin C (2016) " why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 1135–1144
16. Wallkötter S, Tulli S, Castellano G, Paiva A, Chetouani M (2021) Explainable embodied agents through social cues: a review. ACM Trans Hum Robot Inter 10(3):1–24

17. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-cam: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp 618–626
18. Doshi-Velez F, Kim B (2017) Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608
19. Mittelstadt B, Russell C, Wachter S (2019) Explaining explanations in AI. In: Proceedings of the Conference on Fairness, Accountability, and Transparency, pp 279–288
20. Hein D, Udluft S, Runkler TA (2018) Interpretable policies for reinforcement learning by genetic programming. Eng Appl Artif Intell 76:158–169
21. Skirzyński J, Becker F, Lieder F (2021) Automatic discovery of interpretable planning strategies. Mach Learn, 1–43
22. Silva A, Gombolay M, Killian T, Jimenez I, Son S-H (2020) Optimization methods for interpretable differentiable decision trees applied to reinforcement learning. In: International Conference on Artificial Intelligence and Statistics, pp 1855–1865 . PMLR
23. Paleja R, Chen L, Niu Y, Silva A, Li Z, Zhang S, Ritchie C, Choi S, Chang KC, Tseng HE, et al (2023) Interpretable reinforcement learning for robotics and continuous control. arXiv preprint arXiv:2311.10041
24. Booth S, Zhou Y, Shah A, Shah J (2020) Bayes-trex: a bayesian sampling approach to model transparency by example. arXiv preprint arXiv:2002.10248
25. Verma S, Dickerson J, Hines K (2020) Counterfactual explanations for machine learning: a review. arXiv preprint arXiv:2010.10596
26. Zahedi Z, Olmo A, Chakraborti T, Sreedharan S, Kambhampati S (2019) Towards understanding user preferences for explanation types in model reconciliation. In: 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp 648–649 . IEEE
27. Dodeja L, Tambwekar P, Hedlund-Botti E, Gombolay M Towards the design of user-centric strategy recommendation systems for collaborative human-AI tasks. Int J Hum Comput Stud, 103216
28. Grice HP (1975) Logic and conversation. In: Speech acts, pp 41–58. Brill
29. Briggs G, Scheutz M (2011) Facilitating mental modeling in collaborative human-robot interaction through adverbial cues. In: Proceedings of the SIGDIAL 2011 Conference, pp 239–247
30. Sanneman L, Shah JA (2023) Validating metrics for reward alignment in human-autonomy teaming. Comput Hum Behav 146:107809
31. Boggess K, Kraus S, Feng L (2022) Toward policy explanations for multi-agent reinforcement learning. arXiv preprint arXiv:2204.12568
32. Kaupp T, Makarenko A, Durrant-Whyte H (2010) Human-robot communication for collaborative decision making-a probabilistic approach. Robot Auton Syst 58(5):444–456
33. Sanneman L, Tucker M, Shah J (2023) An information bottleneck characterization of the understanding-workload tradeoff. arXiv preprint arXiv:2310.07802
34. Luebbers M, Tabrez A, Ruvane K, Hayes B (2023) Autonomous justification for enabling explainable decision support in human-robot teaming. In: Proceedings of Robotics: Science and Systems, Daegu, Republic of Korea. https://doi.org/10.15607/RSS.2023.XIX.002
35. Andre D, Russell SJ (2002) State abstraction for programmable reinforcement learning agents. In: AAAI/IAAI, pp 119–125
36. Abel D, Arumugam D, Lehnert L, Littman M (2018) State abstractions for lifelong reinforcement learning. In: International Conference on Machine Learning, pp 10–19
37. Fikes RE, Nilsson NJ (1971) Strips: a new approach to the application of theorem proving to problem solving. Artif Intell 2(3–4):189–208
38. Sadigh D, Dragan AD, Sastry S, Seshia SA (2017) Active preference-based learning of reward functions
39. Bellman R (1957) A markovian decision process. J Math Mech 679–684
40. Hoffman G, Bhattacharjee T, Nikolaidis S (2023) Inferring human intent and predicting human action in human–robot collaboration. Ann Rev Contr Robot Auton Syst **7**
41. Korte BH, Vygen J, Korte B, Vygen J (2011) Combinatorial optimization vol. 1. Springer
42. "Gurobi Optimization LLC": Gurobi Optimizer (2019). http://www.gurobi.com
43. McCluskey EJ (1956) Minimization of Boolean functions. Bell Syst Tech J 35(6):1417–1444
44. Sanneman L, Shah JA (2022) An empirical study of reward explanations with human-robot interaction applications. IEEE Robot Autom Lett 7(4):8956–8963
45. Hoffman RR, Mueller ST, Klein G, Litman J (2018) Metrics for explainable AI: Challenges and prospects. arXiv preprint arXiv:1812.04608
46. Cruz F, Young C, Dazeley R, Vamplew P (2022) Evaluating human-like explanations for robot actions in reinforcement learning scenarios. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 894–901. IEEE
47. Faul F, Erdfelder E, Lang A-G, Buchner A (2007) G* power 3: a flexible statistical power analysis program for the social, behavioral, and biomedical sciences. Behav Res Methods 39(2):175–191

48. OpenAI: GPT-4 Technical Report (2023)
49. Ehsan U, Tambwekar P, Chan L, Harrison B, Riedl MO (2019) Automated rationale generation: a technique for explainable AI and its effects on human perceptions. In: Proceedings of the 24th International Conference on Intelligent User Interfaces, pp 263–274
50. Shin D (2021) The effects of explainability and causability on perception, trust, and acceptance: implications for explainable AI. Int J Hum Comput Stud 146:102551
51. Jian J-Y, Bisantz AM, Drury CG (2000) Foundations for an empirically determined scale of trust in automated systems. Int J Cogn Ergon 4(1):53–71
52. Schaefer KE (2016) Measuring trust in human robot interactions: development of the "trust perception scale-hri". In: Robust intelligence and trust in autonomous systems, pp 191–218. Springer
53. Hart SG, Staveland LE (1988) Development of nasa-tlx (task load index): results of empirical and theoretical research. In: Advances in psychology, vol 52, pp 139–183. Elsevier
54. Chang CT, Luebbers MB, Hebert M, Hayes B (2023) Human non-compliance with robot spatial ownership communicated via augmented reality: implications for human-robot teaming safety. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp 9785–9792. IEEE
55. Wagner AR, Borenstein J, Howard A (2018) Overtrust in the robotic age. Commun ACM 61(9):22–24
56. De Visser EJ, Peeters MM, Jung MF, Kohn S, Shaw TH, Pak R, Neerincx MA (2020) Towards a theory of longitudinal trust calibration in human-robot teams. Int J Soc Robot 12(2):459–478
57. Tabrez A, Luebbers MB, Hayes B (2022) Descriptive and prescriptive visual guidance to improve shared situational awareness in human-robot teaming. In: Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, pp 1256–1264
58. Paleja R, Silva A, Chen L, Gombolay M (2020) Interpretable and personalized apprenticeship scheduling: Learning interpretable scheduling policies from heterogeneous user demonstrations. Adv Neural Inf Process Syst 33:6417–6428
59. Garrido-Jurado S, Muñoz-Salinas R, Madrid-Cuevas FJ, Marín-Jiménez MJ (2014) Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recogn 47(6):2280–2292
60. Nikolaidis S, Zhu YX, Hsu D, Srinivasa S (2017) Human-robot mutual adaptation in shared autonomy. In: 2017 12th ACM/IEEE International Conference on Human-Robot Interaction HRI, pp 294–302. IEEE
61. Kwon M, Biyik E, Talati A, Bhasin K, Losey DP, Sadigh D (2020) When humans aren't optimal: robots that collaborate with risk-aware humans. In: Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction, pp 43–52
62. Jeon HJ, Milli S, Dragan AD (2020) Reward-rational (implicit) choice: A unifying formalism for reward learning. arXiv preprint arXiv:2002.04833
63. Miller T (2019) Explanation in artificial intelligence: insights from the social sciences. Artif Intell 267:1–38
64. Millecamp M, Htun NN, Conati C, Verbert K (2019) To explain or not to explain: the effects of personal characteristics when explaining music recommendations. In: Proceedings of the 24th International Conference on Intelligent User Interfaces, pp 397–407
65. Silva A, Tambwekar P, Schrum M, Gombolay M (2024) Towards balancing preference and performance through adaptive personalized explainability. In: Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction, pp 658–668

Authors and Affiliations

Aaquib Tabrez[1,2] · Ryan Leonard[1] · Bradley Hayes[1]

✉ Aaquib Tabrez
tabrez@usc.edu

Ryan Leonard
ryan.leonard@colorado.edu

Bradley Hayes
bradley.hayes@colorado.edu