Towards Collaborative BDI Agents for Human-AI Teamwork

Margarita H. Radeva and Rafael C. Cardoso[0000−0001−6666−6954]

University of Aberdeen, Aberdeen, United Kingdom
margarita.radeva@abdn.ac.uk, rafael.cardoso@abdn.ac.uk

Abstract. This paper presents the development of Belief-Desire-Intention (BDI) agents for human-AI teamwork in the Overcooked AI environment, and investigates their effectiveness and transparency as collaborative partners. The proposed agent architecture explicitly models beliefs, goals, and plans, enabling dynamic adaptation of strategies in response to human actions and evolving environmental conditions. Results from a controlled user study demonstrate that the BDI agent achieves superior collaboration and greater transparency compared with state-of-the-art Deep Reinforcement Learning agents. Participants perceived the agent's behaviour as more predictable and its intentions as more readily interpretable, thereby supporting enhanced mutual understanding in collaborative tasks.

Keywords: BDI Agents · Human-AI Teamwork · Collaborative Agent · Overcooked AI.

1 Introduction

Human–AI collaboration is an increasingly important research area as intelligent agents are expected to work alongside humans in complex tasks [19]. Recent work has emphasised the need to rethink collaborative paradigms to fully leverage AI support in human teams [11,19].

Symbolic AI approaches (e.g., rule-based agents) represent knowledge using symbolic representations of the environment, encoding information as discrete symbols. These representations form the basis of symbolic agents that can interpret them, reason over them, and act accordingly [22]. Symbolic agents based on the Belief-Desire-Intention (BDI) model [4,18] support explicitly structured beliefs, goals, and plans that make their behaviour interpretable. Such agents can be aligned with human reasoning by encoding domain knowledge and task priorities in a transparent, explainable manner [23,21].

Cooperative benchmarks such as *Overcooked AI* [6] have been proposed to study human–AI coordination in a dynamic game setting [6]. However, the current Overcooked AI agents utilise Deep Reinforcement Learning (DRL) models that, while achieving high scores in self-play, suffer from several limitations [6,7,15]. These DRL agents are opaque, lack interpretability, and often

fail to generalise well outside the context in which they were trained [14]. Moreover, DRL agents do not explicitly model human-like reasoning or allow transparent explanations of their decisions [20,14]. By contrast to DRL, BDI agents can adapt their plans during execution and can explain or inspect their decision process [5,1]. Furthermore, BDI agents enable dynamic role allocation through intention selection informed by perceptual beliefs and proactive plan selection informed by human actions. These characteristics make them well-suited to human-AI collaboration, where mutual understanding and teamwork are critical. In particular, we implement our BDI agent using Jason [3], an agent-oriented programming language based on AgentSpeak(L) that enables the development of intelligent agents following the BDI architecture.

This paper addresses a key research gap by evaluating the performance of a BDI agent in real-time cooperation with humans, specifically within the *Overcooked AI* benchmark. Our contribution is a systematic framework[1] for collaborative behaviour in Overcooked AI that explicitly reasons about spatial constraints, shared tasks, and plan selection. We evaluate our work by comparing the resulting BDI agent against established DRL baselines through a controlled user study. The results indicate that the BDI agent provides stronger collaborative performance and improved transparency.

2 The Overcooked AI Environment

Overcooked is a "chaotic cooperative cooking game" where one to four players can work together as chefs to prepare and serve various orders under a strict time limit of 90 seconds. In each level, players must develop their collaborative skills by coordinating tasks such as gathering and chopping ingredients, cooking dishes (e.g., burgers), and occasionally washing used dishes. The game is fully collaborative, meaning all players benefit from working as a team, sharing the same set of orders to cook and the same score. Players are presented with a list of orders that no individual player can handle on their own within the 90-second time limit.

To study human-AI collaboration in a controlled setting, [6] developed *Overcooked AI*, a simplified benchmark environment based on the *Overcooked* game. *Overcooked AI* abstracts the core cooperative cooking task into a discrete grid-world environment suitable for AI training and evaluation. The environment maintains the essential cooperative structure while simplifying mechanics: two agents (players) in a kitchen must work together to cook and serve dishes. Notably, [6] focuses on a single recipe, an onion soup, and a few object types (onion and plate dispensers, pots, serving stations) while preserving low-level motion coordination challenges and high-level strategy challenges [6]. Figure 1 shows a typical *Overcooked AI* level.

Agents in *Overcooked AI* have a discrete action space of six actions. `UP` to move one tile upward (north), the player's y-coordinate decreases by 1. `DOWN` to

[1] Modified *Overcooked AI*: https://github.com/margaritaradeva/OvercookedAI
BDI agent: https://github.com/margaritaradeva/SymbolicAIAgent

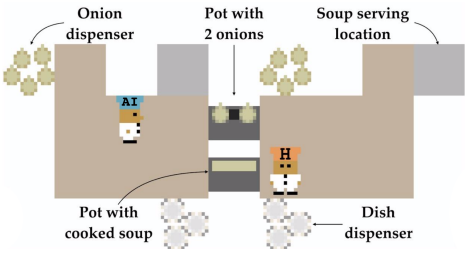


Fig. 1: An example of an *Overcooked AI* level [6] with a discrete grid-world kitchen with onion and plate dispensers, a cooking pot and a serving location.

move one tile downward (south), the player's y-coordinate increases by 1. `LEFT` to move one tile left (west), the player's x-coordinate decreases by 1. `RIGHT` to move one tile right (east), the player's x-coordinate increases by 1. `INTERACT` is context-dependent "pick up/drop/cook" with the object or tile the player is facing (e.g., pick up ingredient, place ingredient in pot, fill plate, serve dish). `NO-OP` the player remains in place and does not interact. At each timestep, an agent can move in one of four directions or interact with the environment. These primitive actions form the building blocks of every interaction in *Overcooked AI*.

The game environment in *Overcooked AI* is much simpler than the original. No chopping of ingredients or burning of food is incorporated in the environment's mechanics. Additionally, although orders are presented as a list, there is no requirement (or bonus) to serve them in the order they are listed. In the current version of *Overcooked AI*, it is also possible for players to produce an incorrect dish. For example, cooking only two onions when three are required. There is no 'trash' action to discard such a dish, and the malformed soup must be left on any free counter tile. We modified the game environment to allow things to be discarded at the serving location. The game interface displays the list of pending orders and the remaining time at the bottom of the screen, but provides no explicit visual cue (e.g., a dish popping out of the list) when an order is completed. We have also added this functionality to enhance the experience for human players, making it more visually appealing and informative.

In [6], an evaluation of human–AI collaboration in the *Overcooked AI* benchmark is introduced. They collect human–human gameplay data and use Behaviour Cloning (BC) to build a proxy model of human behaviour. Although imperfect, this BC model mimics a typical human player while training other agents. Building on this human proxy, the authors train and compare several DRL agents using different techniques, primarily variants of Proximal Policy Optimisation (PPO) and Population-Based Training (PBT). Across both simulation and human-user studies, the human-aware PPO agent achieves the best collaborative performance, significantly outperforming other agents by effectively adapting to human behaviour. The human-aware PPO agent is a PPO agent fine-tuned to maximise rewards when paired with a BC human model, effectively learning a best-response to human-like play.

3 Related Work

Several papers have explored different aspects of human–AI coordination in *Overcooked AI*. For example, in [7], the authors address the problem of partner overfitting, where an agent adapts to its training teammates. They find that common approaches often produce agents that generalise poorly to unfamiliar partners. Consequently, in [7], they include diverse pre-trained partner agents throughout training, yielding agents with more robust coordination strategies and thereby improving teamwork in the environment. Although their work is based on DRL, it offers a clear lesson for symbolic agents: designing for partner diversity can help avoid unstable and over-specialised behaviours.

Another paper proposes a unit-testing framework [15] that highlights the need for robustness in realistic human–AI interactions. The authors construct specific test scenarios to probe agent behaviour under rare but critical situations, such as when the human partner behaves unexpectedly. This approach uncovers edge-case failures that the previous approaches miss. An insight from this study, applicable to symbolic agent design, is the value of building in explicit failure plans and recovery behaviours so that the agent can safely handle unexpected human actions. However, a limitation across these works is that all evaluations remain simulation-only, leaving real-world performance with human collaborators untested [7,15].

To address this gap, [16] conducts an empirical study in which 43 human players team up with two types of DRL agents. The second agent is trained for collaboration by incorporating a model of human behaviour, leading to significantly better team outcomes, with participants serving more dishes on average. Moreover, participants rated the quality of the interaction higher with this agent. These results align with the conclusions of [6], indicating that agents that adapt to human interaction patterns perform more effectively.

The work in [9] shows that task environment plays a critical role in human-AI collaboration. The authors explore this problem by procedurally generating diverse *Overcooked AI* kitchen layouts and examining their effects on cooperation. Through simulations and an online survey, they find that different layouts lead to various team behaviours and collaboration strategies. What works in one environment might fail in another due to changes in task allocation or timing. For symbolic agents, which can reason about goals and spatial configurations, such adaptability can be achieved by incorporating environment-specific plans.

Across these papers, several ideas for effective human-AI collaboration become apparent. Agents must account for human behaviour in their decision-making. Additionally, they should be robust and adaptable to unexpected human behaviours, and consider the constraints of their environment. However, many DRL approaches suffer from opacity, break down outside their training conditions (i.e., are not generalisable), and require large amounts of data [14]. In practice, these large data and computing requirements often translate into longer training times, higher costs and a larger energy (and thus environmental) impact. These limitations motivate an alternative approach using symbolic agents that emphasises transparency, reasoning, flexibility, and sustainability.

4 Collaborative Human-BDI Framework

Our framework adopts a client–server architecture in which interactions among the human user (client), the BDI agent (client), and the *Overcooked AI* environment (server) are coordinated in real time. When the human player interacts with the environment by pressing a key on the keyboard, the *Overcooked AI* frontend captures this input and transmits it to the server. The server forwards this interaction data to the relevant backend logic modules, which update the game state. Once the environment has been updated, the new game state is relayed back to the frontend to visually reflect the changes on the user's screen.

A parallel communication loop occurs for the BDI agent. Rather than relying on keyboard input, the Jason agent sends symbolic actions (e.g. 'UP') directly to the *Overcooked AI* server. These actions are processed in the same manner as those of the human player. Once the game state is updated, it is sent back to the agent, which updates its belief base and determines its next course of action. In contrast to DRL agents, which typically require extensive training and often operate opaquely [17,14], symbolic agents are logic-driven, interpretable and easily modifiable in real time [3]. This design not only enhances the system's transparency and ease of debugging but also enables richer, more meaningful human-agent cooperation [12].

The information flow in our system is illustrated in Figure 2. 1) The human player and BDI agent each send their actions through the frontend or directly to the server, respectively. 2) Human actions are sent from the frontend to the Overcooked AI server. 3) The server forwards all actions to backend modules for processing. 4) and 5) The updated game state is returned to both the frontend (to update the game view for the human) and the BDI agent (to update its belief base). The cycle repeats continuously as new actions are received. This loop enables real-time, bidirectional communication and coordination between agents and the environment.

4.1 BDI Agent

The BDI agent is implemented as a Jason system composed of two primary components: an *AgentSpeak .asl* file and a *Java* environment class (`Kitchen.java`). The *AgentSpeak* encodes the agent's beliefs, goals and plans, while the Java class extends Jason's Environment class to communicate with the *Overcooked AI* game. Perceptual data from *Overcooked AI* (via Socket.IO state updates) is parsed into symbolic beliefs (e.g., agent_position(X,Y) or plate(X,Y)) that populate the agent's belief base. The agent's reasoning then generates intentions based on these beliefs, selects the appropriate plan, and executes the plan's body by sending actions (e.g., *'UP'*, *'DOWN'*) back to the *Overcooked AI* server.

The BDI agent operates autonomously, using its perceptions and beliefs, rather than relying on a central controller or explicit communication with teammates. In the *Overcooked AI* scenario, the BDI agent treats the Kitchen environment and the human player as part of its perceptual input, but all coordination strategies are derived from reasoning over plans in the agent's plan library.

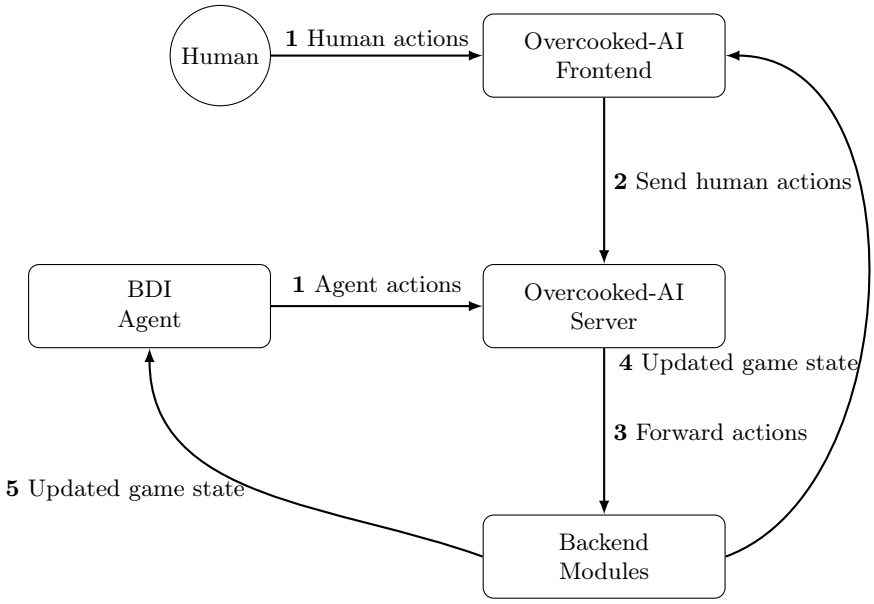


Fig. 2: Continuous real-time loop of actions and state updates.

The beliefs are derived from three main sources:

- *java_layout:* received once at the start of the game to provide static world information such as map layout, terrain features and initial beliefs such as positions of objects. It provides the kitchen layout encoded as ASCII characters (e.g., `P` for pot dispensers, `T` for tomato dispensers), along with initial parameters such as player positions, game duration, and other environment constants.
- *java_state_update:* received every 0.2 seconds[2] to provide dynamic perceptual updates, including player positions, object states, item placements, orders list updates and others.
- *Internal derivation rules:* used to compute higher-level abstractions or infer relational/spatial facts, such as proximity, adjacency and reachability, from perceptual data.

The BDI agent's belief base is populated with predicates derived from environment percepts. Most beliefs come directly from state updates in *java_layout* or *java_state_update*. For example, *onion(X,Y)* denotes an onion dispenser at location (X,Y) and *agent_position(X,Y)* denotes the agent's current grid coordinates. Other beliefs, such as reachability (*can_reach_agent(...)*), are computed using the A* pathfinding algorithm. These beliefs enable the agent to maintain an internal model of the world and reason accordingly.

[2] This was the optimal value we found through experimentation, but it can be customised.

The BDI agent uses external actions to affect the *Overcooked AI* game state and internal actions to compute or modify beliefs within its own belief base. In addition, it uses belief base rules defined directly in *AgentSpeak* to perform symbolic inference over known beliefs.

External actions produce observable effects in the environment, such as movement, picking up or dropping an object, or delivering a soup. These actions are translated into symbolic commands (e.g., *LEFT*, *RIGHT*) by the *Kitchen.java* environment class and sent to the *Overcooked AI* backend via *Socket.IO*. From the perspective of the game engine, these actions are indistinguishable from those of a human player.

Internal actions are invoked from within *AgentSpeak* plans but are implemented in *Kitchen.java* (or separate Java files) as methods that perform additional computation or manipulate the agent's belief base. For instance, the internal action `compute_path(X,Y)` triggers an A* search in Java to generate a movement sequence toward a destination.

Belief base rules are declarative constructs used to infer new facts from existing beliefs. For example, the rule `countInRecipe(Ingredient, Recipe, N)` calculates how many times an ingredient of a given type (e.g., onion) appears in a given recipe. This is particularly useful when reasoning about symbolic structures, such as recipe goals or plan dependencies.

4.2 Spatial Reasoning

To navigate the grid-based kitchen, the agent uses A* pathfinding implemented in Java. The *Kitchen.java* class maintains a 2D grid, marking free and blocked cells based on `terrain(X,Y,Type)` percepts (e.g., counters, dispensers and obstacles are marked as blocked).

When the agent needs to move to the target coordinates, it calls the internal action `compute_path(goalX, goalY)`. This triggers a method that rebuilds the grid from percepts, calls the A* search and translates the resulting coordinate path into *Overcooked AI* movement actions. A simplified snippet of the `go_to` plan is shown in Listing 1.1. This plan attempts to move the agent toward an object. The call to `compute_path(Z,M)` causes the environment to run A* and emit the low-level move actions. Once movement concludes, a subgoal `!turn_to(Object)` ensures the agent is facing the object before interaction.

Listing 1.1: A simplified snippet of the go_to plan invoking A* pathfinding.

```
1 +!go_to(Object) : agent_position(X,Y) & object(Object,Z,M)
      & not adjacent(X,Y,Z,M)
2 <-
3 compute_path(Z,M);
4 !go_to(Object).
```

Notably, invoking the A* pathfinding logic via an internal action does not disrupt the agent's current intention. In Jason's BDI architecture, internal actions are treated as regular steps in the plan body. They execute and return

before control proceeds to the next step. As a result, the BDI agent maintains its current goal structure and intention stack. Pathfinding is offloaded to the environment as a discrete, non-blocking calculation, allowing the agent to retain full autonomy over its decision-making process while delegating movement complexity to a lower abstraction layer.

Repeated path recalculations and goal persistence handle collision avoidance for the agent. The BDI agent does not blindly follow a fixed path, since the human player can move unpredictably and may block the agent. Instead, each time the agent attempts to execute a movement plan (e.g., going for a plate), it requests a new A* path based on the current *Overcooked AI* environment state, which includes the human's latest position (`human_position(X,Y)`). If the path is valid, the agent begins to follow it.

For instance, if the human steps into the BDI agent's planned path, the agent's next move may fail or become invalid (the tile is now occupied). In cases like these, the movement action has no effect on the environment, and the plan fails 'silently' within the reasoning cycle (via failure plans). The BDI agent will automatically attempt the goal again by triggering the appropriate event and rerunning A* based on the updated beliefs. This loop continues until a valid new path is found and completed (e.g., the agent goes 'around' the human player). If no such path is available, the agent waits until one becomes available.

4.3 Task Reasoning

Four reasoning behaviours are created: 1) *single reachability*, where the agent can access *only one category* of objects (e.g., only soup ingredients) and the human player can reach the other three (pots, plates, serving stations); 2) *double reachability*, where the agent and human each have access to two distinct object types; 3) *triple reachability*, where the agent reaches three different categories and the human player reaches only one; and 4) *full reachability*, where both players have full access to all objects.

Overlapping cases, where both players can reach certain stations, such as the BDI agent having access to pots and ingredients only, and the human player having access to pots, plates, and serving stations, are handled automatically by the agent's belief base rules. A plan requiring `cannot_reach_agent(pot, X, Y)` will not activate if the agent can reach a pot. To check these reachabilities, the Jason environment runs A* on initialisation and adds the appropriate beliefs (e.g., `can_reach_agent(obj, X, Y)` and `cannot_reach_agent(obj, X, Y)`).

The asymmetric setups (1–3) employ forced coordination. No single player can complete all soup-making steps alone (i.e., an agent cannot make a soup on its own if it cannot reach all ingredients). This design means the BDI agent must deliberately choose which subtasks to perform and which not to. It also demonstrates the modularity of the agent's reasoning. A simplified example of the agent's behaviour under each reachability condition is presented as follows (not all cases are described):

- *Single reachability:* Assume the BDI agent can reach only plates while the human player can reach ingredients, pots and serving locations. The agent

waits for the soup to start cooking, picks up a plate and hands it to the human player at a shared counter so the soup can be plated and served. This cycle repeats until all orders are complete.
- *Double reachability:* Assume the BDI agent can reach plates and pots, and the human player has access to ingredients and serving locations. If the human passes an ingredient that matches a recipe in the order list and the pot's existing contents (a pot can be empty), the agent adds it to that pot. Otherwise, the agent places the ingredient on a private counter to clear the shared space (if valid for a recipe, the agent will use it later). When a soup is cooked, the agent picks up a plate, plates the soup and passes it to the human for serving, freeing space first if needed. If no ingredients are available, the agent waits. This cycle repeats until all orders are complete.
- *Triple reachability:* This scenario is similar to the previous, except the BDI agent can now access three categories of objects, giving it more options for which tasks to perform.
- *Full reachability:* Both players can reach all objects. The agent follows the human's actions and adapts to them. If the human begins cooking a second order, the agent helps load its ingredients. If the human adds the wrong ingredient, the agent finishes cooking and serves it for zero points to clear the cooking pot. When serving, if the human is closer to the plate dispenser, the BDI agent defers, and if the human does not pick up the plate in time, the agent serves the soup itself. Should the human arrive at a station before the agent (for example, the pot starts cooking while the agent is still holding an ingredient), it places its item on an empty counter. Likewise, if the agent intends to serve but the human also decides to do so, the agent adjusts its plan accordingly.

One of the key innovations in the agent's design is its ability to adopt different coordination roles depending on the collaborative tasks at hand. When the kitchen layout restricts the agent to accessing only one or two object categories (e.g., ingredients), the agent typically assumes a 'leader' role. In these cases, it decides the order in which ingredients are needed and delivers them to the human using the shared counters. The human, with access to pots, plates, or serving stations, takes on a 'follower' role, completing the tasks.

Conversely, when the BDI agent is limited to accessing only certain types of objects (e.g., pots or plates), it acts as a 'follower'. Here, it waits for the human to pass along the necessary ingredients before continuing the process to complete a recipe. This adaptive behaviour emerges naturally from the BDI model. The agent's beliefs encode the environment's state and what it can reach and act upon, while its intentions are constructed from plans conditioned on those beliefs.

Central to this adaptive behaviour are the contextual conditions under which the agent's plans operate. The agent evaluates applicability by consulting its beliefs. For instance, when the BDI agent can only reach plates, it commits to passing them to the human once the soup is ready. This plan is illustrated in Listing 1.2, which presents a brief excerpt of the plan. In this plan, the context

ensures that it can reach plates but not pots or serving stations, that it has at least one pending recipe, and that there is a pot with a ready soup that it cannot reach. When all these conditions hold, the plan's body is executed. It logs a message (in the MAS console), issues the subgoal of passing a plate to the human, and then re-adopts the `can_reach_plates_only` goal. This plan is recursive and will repeat until no recipes remain.

Listing 1.2: Example plan for when the agent can only reach plates.

```
+!can_reach_plates_only : can_reach_agent(plate,X,Y) &
    recipes([FirstRecipe|_]) & pot_ready(A,B) &
    cannot_reach_agent(pot,A,B) & cannot_reach_agent(
    serving,C,D)
<-
    !pass_to_human(plate,1);
    !can_reach_plates_only.
```

These plans are selected through Jason's reasoning cycle. When a relevant event is triggered (e.g., a new goal or an updated perception), the agent filters plans with matching context, ranks them, and commits to one. The modularity of the plans ensures clean fallback. If the agent's role or reachability changes, the context of previously relevant plans fails, and other plans are adopted instead. No retraining is required, unlike in DRL-based systems.

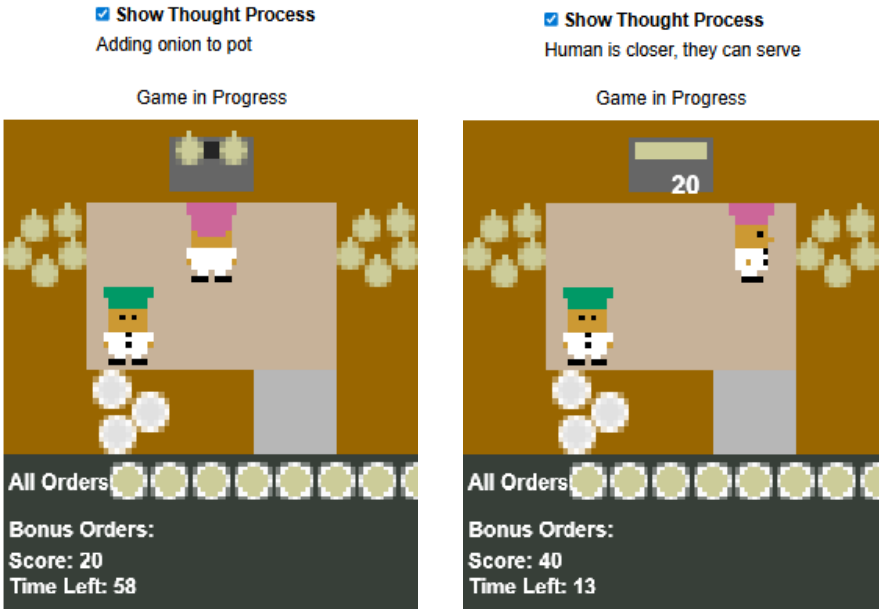
4.4 BDI Thought Bubble

A custom internal action, `thought`, is created in the `Kitchen.java` environment class. This action enables the BDI agent to emit plain-text strings representing its current intention. These messages are sent to the *Overcooked AI* server using a newly introduced `thought` Socket.IO event and are rendered on the screen as visual overlays during gameplay.

For example, before adopting a subgoal such as `!go_to(plate)`, the corresponding plan includes a statement such as `thought("I'm going to grab a plate")`, which is then shown to the player as simple textual output on the screen in real time. The 'thought' changes dynamically as the agent acts. The output is controlled by a checkbox in the interface, allowing players to enable or disable thought messages as they prefer.

In Figure 3, we show two examples of the thought bubble in practice. Figure 3a displays the BDI agent adding onions to the pot to cook an onion soup, and the thought bubble relays that information to the human as "Adding onion to pot". The thought bubble in Figure 3b identifies that the human is closer to the plates. Therefore, the BDI agent informs the human to collect the plate, serve the dish from the pot, and deliver it.

Beyond technical benefits, this feature also addresses another issue in human-AI collaboration. When humans become aware that their partner is an agent, they often reduce their own level of cooperation, expecting the agent to bear the burden.



(a) Agent adding onions. (b) Agent deferring to human.

Fig. 3: Two examples of the thought bubble in the *Cramped Room* layout. On the left a 'leader' behaviour and on the right a 'follower' behaviour. The chef with a green hat (on the left) is the human and the other (with a purple hat) is the BDI agent.

5 Evaluation

A user study using a standard usability questionnaire is employed to quantify and analyse the agent's performance. This study also compares the BDI agent against an established DRL agent, the human-aware PPO agent, identified by [6] as the best-performing DRL model for collaborating with humans in the *Overcooked AI* system.

Comparing symbolic and DRL approaches is essential, as symbolic agents offer interpretable, directly modifiable behaviour, whereas DRL agents often act as black boxes [14]. Therefore, this evaluation investigates whether the theoretical advantages of symbolic agents, such as interpretability and modularity, translate into superior human-agent collaboration and whether any usability issues arise in practice. The findings inform future design refinements, validate the symbolic approach in applied settings, and guide the broader adoption of symbolic methods in human-AI teamwork.

A within-subjects, counterbalanced design is chosen so that each participant plays with both the symbolic and human-aware PPO agents in a randomised order. A custom Python script ensures that across participants, each agent-layout pairing occurs equally often, mitigating both learning and layout-specific biases.

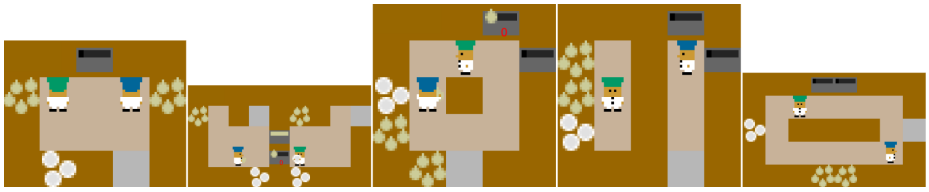


Fig. 4: Overcooked AI layouts [6]. From left to right: *Cramped Room*, *Asymmetric Advantages*, *Coordination Ring*, *Forced Coordination*, and *Counter Circuit*.

Participants were informed which agent controlled the other avatar. This choice was deliberate and intended to preserve the validity of the interaction rather than to bias it. In early pilot sessions, several participants reported confusion about whether the teammate was human or artificial, which affected their behaviour, increased hesitation, and often led to strategic choices that were unrelated to the study objectives. Making the agent's identity explicit avoided this ambiguity and ensured that participants engaged with the teammate as an AI collaborator rather than an unpredictable human teammate. Since the study aimed to assess perceived effectiveness and transparency of specific AI behaviours, it was important that participants attributed those behaviours to the correct system. Our study focuses on qualitative perceptions rather than a fully controlled comparison.

The primary evaluation instrument used is the System Usability Score (SUS), selected for its reliability, brevity and sensitivity to usability differences at small sample sizes [2]. Secondary metrics include task success rate ($\frac{\text{completed orders}}{\text{total orders}}$) to determine collaborative effectiveness. These measures, taken together, reveal usability strengths and weaknesses, and inform areas for improvement.

We used the same five layouts as the human-aware PPO agent in [6] to avoid potential issues with retraining their agent. The layouts are shown in Figure 4. From left to right: *Cramped Room* proposes coordination hurdles in the tight, shared space, as agents can easily bump into one another. *Asymmetric Advantages* evaluates whether players can adopt high-level tactics that leverage their individual strengths. *Coordination Ring* requires partners to work together to move from the lower-left corner to the upper-right corner of the map without bumping into one another. *Forced Coordination* eliminates the risk of collisions but forces players to plan jointly at a high level, because no one can complete an order on their own. *Counter Circuit* is similar to *Coordination Ring*, it tests players' ability to navigate through a corridor without collision.

5.1 Survey Design

The survey, developed using Microsoft Forms, includes both quantitative (SUS scores) and qualitative (open-ended questions) measures to capture both performance metrics and user perceptions. It is structured into the following sections:

- *Participant Background:* Questions on video-gaming experience and familiarity with the original *Overcooked*. Demographic questions are minimised to maintain a clear focus. Responses were anonymised to encourage honest feedback.
- *Agent Collaboration Experience:* Questions assessing perceived effectiveness, ease of collaboration, encountered issues, and overall preference for each agent. Questions are presented in two separate sections and shuffled per participant to reduce response bias.
- *SUS Questionnaire:* A ten-item, five-point Likert scale questionnaire, which provides a reliable, global measure of the participants' perceived usability of the system. Similar to the previous set of questions, the SUS is completed separately for each agent (DRL and BDI), split into two sections and shuffled for each participant.

Due to the complexity of the experiments, we opted for a reduced sample size and recruited 14 participants from a broad pool of students and staff across a wide range of disciplines at several higher education institutions. Demographic questions were deliberately minimised to maintain a focused evaluation and encourage participation. The survey, therefore, prioritised gaming experience and familiarity with Overcooked rather than broader technical background. Participants came from a range of disciplines and were not selected based on expertise in AI or agent-based systems, which aligns with the aim of assessing collaboration from the perspective of typical end users rather than domain specialists.

Each participant completes the *Overcooked AI* tutorial and follows its instructions before playing with the agents. Each participant is asked to: 1) Play the same randomly selected layout with both the BDI agent and the human-aware PPO agent for 45 seconds each. Layouts included *Cramped Room*, *Asymmetric Advantages*, *Coordination Ring*, *Forced Coordination*, and *Counter Circuit* [6], all configured to prepare three-onion soups for fairness (the DRL agent is limited to these orders). 2) Complete the online survey via the provided link or QR code at their own pace.

The 45-second limit is chosen empirically to allow meaningful interaction while avoiding ceiling effects on success rates. Additionally, multiple layouts are selected to mitigate bias associated with a single experimental environment.

5.2 Survey Results

The following metrics were used to analyse the results:

- *Descriptive Statistics:* Overview of central tendency and variability for both SUS scores and success rates.
- *SUS Score Calculation:* Production of a single usability score on a 0–100 scale with established thresholds (e.g., a score above 68 indicates above-average usability).

A total of 14 participants completed the evaluation. The average survey completion time per participant was 14 minutes and 21 seconds. Most respondents

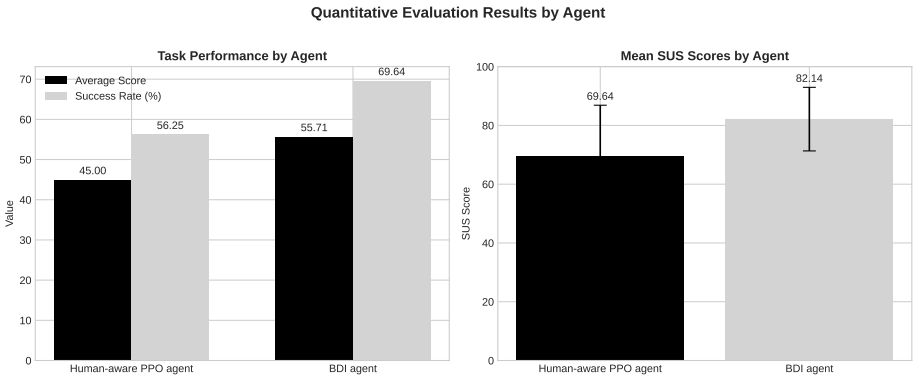


Fig. 5: Results for task performance and mean SUS scores by agent.

were aged 18–24 (85.7%), with one participant each in the 25–34 and 35–44 categories. Gender distribution was predominantly *male (57.1%)*, followed by *female (35.7%)* and one identifying as *other (7.1%)*. Participants were asked to self-assess their video game skills. The majority rated themselves as either *Good* or *Very Good (35.7% each)*, with fewer respondents selecting *Poor (14.3%)*, *Acceptable (7.1%)* and *Very Poor (7.1%)*. Only four participants were exposed to the BDI agent's thought-bubble feature. These sessions were conducted as additional rounds after the main user study, and only free-text feedback was collected afterwards.

Qualitative responses undergo thematic analysis to identify recurring usability issues, unexpected agent behaviours and areas for improvement. Performance was assessed based on average score and task success rate. The SUS scores were gathered separately for each agent. Figure 5 illustrates the quantitative evaluation results obtained for each agent.

Regarding the question "Was the collaboration with the human-aware PPO agent effective?", *71.4%* of participants *agreed* or *strongly agreed*. For the same question regarding the BDI agent, this increased to *85.7%*. Regarding the human-aware PPO agent, *71.5%* of participants *agreed* or *strongly agreed* that the agent focused on scoring points independently. In contrast, only *30.9%* reported the same for the BDI agent, suggesting a better sense of teamwork.

Two additional questions collected participant preferences regarding which agent they perceived as the better collaborator and which was easier to work with. Seven participants (*50%*) considered the BDI agent the better collaborator, three (*21%*) chose the PPO agent, and four (*20%*) stated that they were similar. Ten participants (*71%*) said the BDI agent was easier to work with, three (*21%*) preferred the PPO agent, and one (*7%*) said both were similar.

5.3 Discussion

The results provide evidence that the BDI agent outperforms the human-aware PPO agent in usability and collaboration in the *Overcooked AI* environment. Subjective feedback further reinforced the strengths of the BDI agent. Half of the participants (*50%*) selected the BDI agent as the better collaborator, while only *21%* preferred the human-aware PPO agent. When asked which agent was easier to work with, an even larger majority (*71%*) preferred the BDI agent. These findings suggest a pattern: participants performed better with the BDI agent and also found the interaction more usable, collaborative, and easier to work with.

In addition to quantitative questions, participants were asked to provide textual feedback on their experiences with both agents, offering greater insight into the agents' expected and unexpected behaviour. Several participants commented on the human-aware PPO agent's fast, autonomous behaviour, describing it as "too quick" or acting "on its own" without considering the human player's presence. One participant even noted it felt like the agent was competing with it rather than collaborating. A few also noted that they experienced unexpected behaviour coming from the agent at times when trying to collaborate.

Feedback about the BDI agent was generally more positive, with several participants describing it as "more collaborative", "easier to work with" and "slower but more predictable". Some appreciated that it gave them space to act or to form routines, especially in layouts such as *Coordination Ring*. However, that was not the case for everyone, as a few noted that the agent looks "confused at times", occasionally fails to act efficiently, or is too slow for their preference. These issues suggest a trade-off between transparency and responsiveness.

When asked about behavioural changes over time, most participants noted that both agents were generally consistent in their collaboration. These responses help explain the quantitative trends. The BDI agent was perceived as more cooperative and predictable, aligning with its higher SUS score and stronger participant preference. Conversely, the DRL agent was observed to be faster but more chaotic and less collaborative.

The design of the BDI agent prioritised transparency and structured decision-making, which supports more intuitive human-agent collaboration. Features such as intention-based action selection, spatial awareness and predefined priorities helped create behaviours that users described as predictable and cooperative. This likely contributed to the agent's higher usability and preference ratings. Transparency arises from two features. First, the agent's beliefs and intentions (object locations, teammate position, reachable items, inferred subgoals) directly guide the actions that the human observes. Second, the thought bubble interface displays these intentions in real time. Free-text comments repeatedly described the symbolic agent as predictable, easy to follow, and clear in purpose. Several participants explicitly mentioned the visible subgoals and thought bubbles.

A further constraint was the limited ability to compare the agents across a wider range of layouts while maintaining fairness. The human-aware PPO agent

(as well as the other DRL agents) had been trained only on five specific layouts, and these were the only layouts available for evaluation without retraining.

While the findings are informative, several limitations must be acknowledged. Firstly, the sample size was relatively small ($n = 14$), which reduced the study's potential for statistical analysis and limited the generalisability of the results. The study is presented as an exploratory evaluation that demonstrates feasibility and motivates a future larger-scale experiment. The main contribution is the framework, not statistical generalisation. Additionally, the short interaction time (45 seconds per layout) may not have been sufficient for participants to fully understand or adapt to the agents' behaviour, especially for more complex collaborative dynamics. While these provide valuable insight, they are subjective and may be influenced by individual preferences, familiarity with the task, or momentary frustration. These factors should be taken into account when interpreting the results and drawing broader conclusions on future experiments.

6 Conclusion

This paper introduced a collaborative BDI agent for Human-AI teamwork in the *Overcooked AI* environment. The architecture separates perception, reasoning, and execution. The principles were applied across multiple layouts and object combinations, and the same approach can be adapted to other domains of human-AI teamwork. The results demonstrate that the BDI agent successfully fulfilled the core objectives. It was capable of completing *Overcooked AI* tasks, collaborating effectively with human participants, and achieving higher usability and performance metrics than a state-of-the-art human-aware DRL agent across the five standard layouts. Moreover, participant feedback highlighted the BDI agent's transparency and predictability, traits often lacking in DRL models.

Several promising directions for future research emerge. 1) Deploying *Overcooked AI* to collect results for human-human collaboration trials, allowing agents to be compared directly against natural human coordination as a gold standard. 2) Increasing the number and diversity of participants would allow statistical analysis and strengthen the generalisability of findings. 3) The *Overcooked AI* environment could be extended to include new mechanics such as recipe-specific machines (e.g., juice/ice cream machines), hazards (e.g., burning soups) or moving tiles. These challenges would test the agents' ability to manage more complex coordination strategies and assess their collaborative adaptability under increased pressure, similar to the original Overcooked game. 4) The runtime adaptability of BDI agents can be further explored through dialogue between the BDI agent and the human. Large Language Models (LLMs) have been excelling in this area, and several works have utilised LLMs with BDI agents [8]. ChatBDI [10] translates messages between a BDI agent and a human, from speech acts into natural language and vice versa. NatBDI [13] enables the programming of rational agents in a controlled natural language.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Al Shukairi, H., Cardoso, R.C.: ML-MAS: A hybrid AI framework for self-driving vehicles. In: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems. p. 1191–1199. AAMAS '23, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2023). https://doi.org/10.65109/ETRN2911
2. Benyon, D.: Designing Interactive Systems: A Comprehensive Guide to HCI, UX and Interaction Design. Pearson, Harlow, UK, 3rd edn. (2013)
3. Bordini, R.H., Hübner, J.F., Wooldridge, M.: Programming Multi-Agent Systems in AgentSpeak using Jason. John Wiley & Sons, Chichester, West Sussex, United Kingdom (2007). https://doi.org/10.1002/9780470061848
4. Bratman, M.E., Israel, D.J., Pollack, M.E.: Plans and resource-bounded practical reasoning. Computational Intelligence **4**(4), 349–355 (1988). https://doi.org/10.1111/j.1467-8640.1988.tb00284.x
5. Cardoso, R.C., Ferrando, A.: A review of agent-based programming for multi-agent systems. Computers **10**(2), 16 (2021). https://doi.org/10.3390/computers10020016
6. Carroll, M., Griffiths, T.L., Shah, R., Seshia, S.A., Ho, M.K., Abbeel, P., Dragan, A.: On the utility of learning about humans for human-AI coordination. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS 2019). Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper/2019/hash/f5b1b89d98b7286673128a5fb112cb9a-Abstract.html
7. Charakorn, R., Manoonpong, P., Dilokthanakul, N.: Investigating partner diversification methods in cooperative multi-agent deep reinforcement learning. In: Neural Information Processing. ICONIP 2020. Communications in Computer and Information Science, vol 1333. Springer (2020). https://doi.org/10.1007/978-3-030-63823-8_46
8. Ciatto, G., Aguzzi, G., Battistini, R., Baiardi, M., Burattini, S., Ricci, A.: Exploiting genai for plan generation in bdi agents. In: Proceedings of the 28th European Conference on Artificial Intelligence (ECAI 2025). pp. 3495–3502. IOS Press (2025). https://doi.org/10.3233/FAIA251223
9. Fontaine, M.C., Hsu, Y.C., Zhang, Y., Tjanaka, B., Nikolaidis, S.: On the importance of environments in human-robot coordination. In: Proceedings of Robotics: Science and Systems (RSS 2021) (July 2021). https://doi.org/10.15607/RSS.2021.XVII.038
10. Gatti, A., Mascardi, V., Ferrando, A.: Let me talk to you! natural language interaction between humans and BDI agents via ChatBDI. In: Proceedings of the 28th European Conference on Artificial Intelligence (ECAI 2025). pp. 3646–3654. IOS Press (2025). https://doi.org/10.3233/FAIA251242
11. Gómez, C., Cho, S.M., Ke, S., Huang, C.M., Unberath, M.: Human-AI collaboration is not very collaborative yet: A taxonomy of interaction patterns in AI-assisted decision making from a systematic review. Frontiers in Computer Science **6**, 1521066 (2024). https://doi.org/10.3389/fcomp.2024.1521066
12. Harbers, M., Bradshaw, J.M., Johnson, M., Feltovich, P.J., van den Bosch, K., Meyer, J.C.: Explanation in human-agent teamwork. In: Coordination, Organizations, Institutions, and Norms in Agent System VII, COIN 2011 International Workshops, COIN@AAMAS 2011, Taipei, Taiwan, May 3, 2011, COIN@WI-IAT 2011, Lyon, France, August 22, 2011, Revised Selected Papers. pp. 21–37. Lecture Notes in Computer Science, Springer (2011). https://doi.org/10.1007/978-3-642-35545-5_2

13. Ichida, A.Y., Meneguzzi, F., Cardoso, R.C.: BDI agents in natural language environments. In: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024). pp. 880–888. International Foundation for Autonomous Agents and Multiagent Systems (2024). https://doi.org/10.65109/LGWZ6868
14. Iyer, R., Li, Y., Li, H., Lewis, M., Sundar, S.S., Sycara, K.: Transparency and explanation in deep reinforcement learning neural networks. In: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society (AIES '18). pp. 144–150. Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3278721.3278776
15. Knott, P., Carroll, M., Devlin, S., Ciosek, K., Hofmann, K., Dragan, A.D., Shah, R.: Evaluating the robustness of collaborative agents. In: Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021). pp. 1560–1562. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS) (2021). https://doi.org/10.65109/YNAJ2080
16. Nalepka, P., Gregory-Dunsmore, J.P., Simpson, J., Patil, G., Richardson, M.J.: Interaction flexibility in artificial agents teaming with humans. In: Proceedings of the 43rd Annual Meeting of the Cognitive Science Society (CogSci 2021). pp. 112–118. Cognitive Science Society, Seattle, WA (2021), https://escholarship.org/uc/item/9ks6n70q
17. Ota, K., Jha, D.K., Kanezaki, A.: A framework for training larger networks for deep reinforcement learning. Machine Learning **113**(2), 345–367 (2024). https://doi.org/10.1007/s10994-024-06547-6
18. Rao, A.S., Georgeff, M.: BDI Agents: From Theory to Practice. In: Proc. 1st Int. Conf. Multi-Agent Systems (ICMAS). pp. 312–319. San Francisco, USA (jun 1995)
19. Schmutz, J.B., Outland, N., Kerstan, S., Georganta, E., Ulfert, A.S.: AI-teaming: Redefining collaboration in the digital era. Current Opinion in Psychology **58**, 101837 (2024). https://doi.org/10.1016/j.copsyc.2024.101837
20. Vouros, G.A.: Explainable deep reinforcement learning: State of the art and challenges. ACM Computing Surveys **55**(5), 1–39 (2022). https://doi.org/10.1145/3527448
21. Winikoff, M., Thangarajah, J., Rodriguez, S.: A scoresheet for explainable AI. In: Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems. p. 2171–2180. AAMAS '25, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2025). https://doi.org/10.65109/JXCR5334
22. Wooldridge, M.: An Introduction to MultiAgent Systems. John Wiley & Sons, Chichester, United Kingdom, 2nd edn. (2009)
23. Yan, E., Burattini, S., Hübner, J.F., Ricci, A.: A multi-level explainability framework for engineering and understanding BDI agents. Auton. Agents Multi Agent Syst. **39**(1), 9 (2025). https://doi.org/10.1007/S10458-025-09689-6