Collaborating with Humans without Human Data

DJ Strouse[*], **Kevin R. McKee, Matt Botvinick, Edward Hughes, Richard Everett**[*]
DeepMind
{strouse, kevinrmckee, botvinick, edwardhughes, reverett}@deepmind.com

Abstract

Collaborating with humans requires rapidly adapting to their individual strengths, weaknesses, and preferences. Unfortunately, most standard multi-agent reinforcement learning techniques, such as self-play (SP) or population play (PP), produce agents that overfit to their training partners and do not generalize well to humans. Alternatively, researchers can collect human data, train a human model using behavioral cloning, and then use that model to train "human-aware" agents ("behavioral cloning play", or BCP). While such an approach can improve the generalization of agents to new human co-players, it involves the onerous and expensive step of collecting large amounts of human data first. Here, we study the problem of how to train agents that collaborate well with human partners without using human data. We argue that the crux of the problem is to produce a diverse set of training partners. Drawing inspiration from successful multi-agent approaches in competitive domains, we find that a surprisingly simple approach is highly effective. We train our agent partner as the best response to a population of self-play agents and their past checkpoints taken throughout training, a method we call Fictitious Co-Play (FCP). Our experiments focus on a two-player collaborative cooking simulator that has recently been proposed as a challenge problem for coordination with humans. We find that FCP agents score significantly higher than SP, PP, and BCP when paired with novel agent and human partners. Furthermore, humans also report a strong subjective preference to partnering with FCP agents over all baselines.

1 Introduction

Generating agents which collaborate with novel partners is a longstanding challenge for Artificial Intelligence (AI) [4, 16, 37, 52]. Achieving ad-hoc, zero-shot coordination [31, 66] is especially important in situations where an AI must generalize to novel human partners [6, 61]. Many successful approaches have employed human models, either constructed explicitly [14, 35, 53] or learnt implicitly [12, 60]. By contrast, recent work in competitive domains has shown that it is possible to reach human-level using model-free reinforcement learning (RL) without human data, via self-play [8, 9, 63, 64]. This begs the question: Can model-free RL without human data generate agents that can collaborate with novel humans?

We seek an answer to this question in the space of common-payoff games, where all agents work towards a shared goal and receive the same reward. Self-play (SP), in which an agent learns from repeated games played against copies of itself, does not produce agents that generalize well to novel co-players [10, 11, 21, 44]. Intuitively, this is because agents trained in self-play only ever need to coordinate with themselves, and so make for brittle and stubborn collaborators with new partners who act differently. Population play (PP) trains a population of agents, all of whom interact with each other [39]. While PP can generate agents capable of cooperation with humans in competitive team games [34], it still fails to produce robust partners for novel humans in pure common-payoff settings

[*]Equal contribution.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

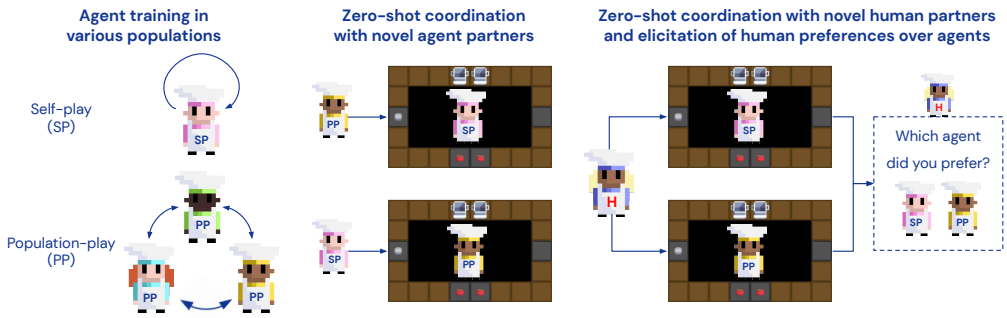


Figure 1: In this work, we evaluate a variety of agent training methods (Section 2) in zero-shot coordination with agents (Section 4). We then run a human-agent collaborative study designed to elicit human preferences over agents (Section 5).

[12]. PP in common-payoff settings naturally encourages agents to play the same way, reducing strategic diversity and producing agents not so different from self-play [24].

Our approach starts with the intuition that the key to producing robust agent collaborators is exposure to diverse training partners. We find that a surprisingly simple strategy is effective in generating sufficient diversity. We train N self-play agents varying only their random seed for neural network initialization. Periodically during training, we save agent "checkpoints" representing their strategy at that point in time. Then, we train an agent partner as the best-response to both the fully-trained agents and their past checkpoints. The different checkpoints simulate different skill levels, and the different random seeds simulate breaking symmetries in different ways. We refer to this agent training procedure as **Fictitious Co-Play (FCP)** for its relationship to fictitious self-play [7, 27, 28, 69].

We evaluate FCP in a fully-observable two-player common-payoff collaborative cooking simulator. Based on the game Overcooked [25], it has recently been proposed as a coordination challenge for AI [12, 50, 70]. State-of-the-art performance in producing agents capable of generalization to novel humans was achieved in [12] via behavioral cloning (BC) of human data. More precisely, BC was used to produce models that can stand in as human proxies during training in simulation, a method we call behavioral cloning play (BCP). We demonstrate that FCP outperforms BCP in generalizing to both novel agent and human partners, and that humans express a significant preference for partnering with FCP over BCP. Our method avoids the cost and potential privacy concerns of collecting human data for training, while achieving better outcomes for humans at test time.

We summarize the novel contributions of this paper as follows:

1. We propose Fictitious Co-Play (FCP) to train agents capable of zero-shot coordination with humans (Section 2.1).
2. We demonstrate that FCP agents generalize better than SP, PP, and BCP in zero-shot coordination with a variety of held-out agents (Section 4.2).
3. We propose a rigorous human-agent interaction study with behavioral analysis and participant feedback (Section 5.1).
4. We demonstrate that FCP significantly outperforms the BCP state-of-the-art, both in task score and in human partner preference (Section 5.2).

2 Methods

2.1 Fictitious Co-Play (FCP)

Diverse training conditions have been shown to make agents more robust, from environmental variations (i.e. domain randomization [54, 56, 67]) to heterogeneity in training partners [69]. We seek to train agents that are robust partners for humans in common-payoff games, and so extend this line of work to that setting.

One important challenge in collaborating with novel partners is dealing with symmetries [31]. For example, two agents A and B facing each other may move past each other by A going left and B going right, or vice versa. Both are valid solutions, but a good agent partner will adaptively switch between

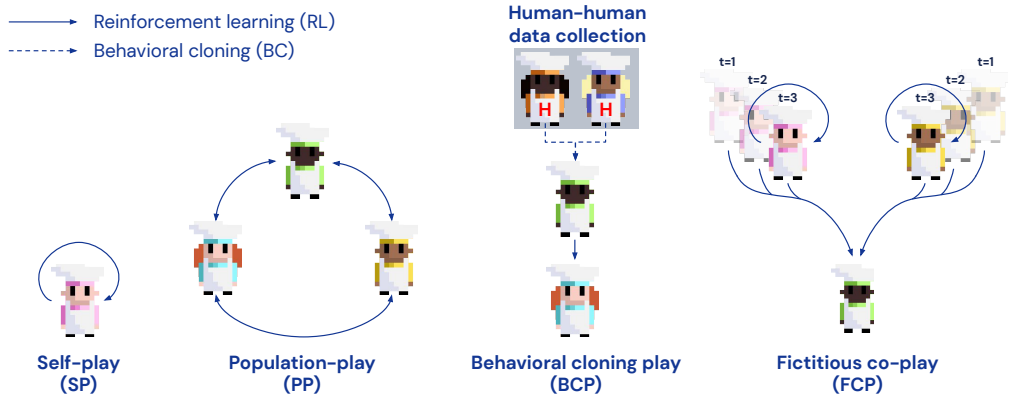


Figure 2: The four agent training methods we evaluate in this work. **Self-play (SP)** where an agent learns with itself, **population-play (PP)** where a population of agents are co-trained together, and **behavioral cloning play (BCP)** where data from human games is used to create a behaviorally cloned agent with which an RL agent is then trained. In our method, **Fictitious Co-Play (FCP)**, N self-play agents are trained independently and checkpointed throughout training. An agent is then trained to best respond to the entire population of SP agents and their checkpoints.

these conventions if a human clearly prefers one over the other. A second important challenge is dealing with variations in skill level. Good agent partners should be able to assist both highly-skilled partners, as well as partners who are still learning.

Fictitious co-play (FCP) is a simple two-stage approach for training agents that overcomes both of these challenges (Figure 2, right). In the first stage, we train a diverse pool of partners. To allow the pool to represent different symmetry breaking conventions, we train N partner agents in self-play. Since these partners are trained independently, they can arrive at different arbitrary conventions for breaking symmetries. To allow the pool to represent different skill levels, we use multiple checkpoints of each self-play partner throughout training. The final checkpoint represents a fully-trained "skillful" partner, while earlier checkpoints represent less skilled partners. Notably, by using multiple checkpoints per partner, this additional diversity in skill incurs no extra training cost.

In the second stage, we train an FCP agent as the best response to the pool of diverse partners created in the first stage. Importantly, the partner parameters are frozen and thus FCP must learn to adapt to partners, rather than expect partners to adapt to it. In this way, FCP agents are prepared to follow the lead of human partners, and learn a general policy across a range of strategies and skills. We call our method "fictitious" co-play for its relationship to fictitious self-play in which competitive agents are trained with past checkpoints (in that case, to avoid strategy cycling) [7, 27, 28, 39, 69].

2.2 Baselines and ablations

We compare FCP agents to the three baseline training methods listed below, each varying only in their set of training partners, with the RL algorithm and architecture consistent across all agents:

1. Self-play (SP), where agents learn solely through interaction with themselves.
2. Population-play (PP), where a population of agents are co-trained through random pairings.
3. Behavioral cloning play (BCP), where an agent is trained with a BC model of a human [12].

We also evaluate three variations on FCP to better understand the conditions for its success:

1. To test the importance of including past checkpoints in training, we evaluate an ablation of FCP in which agents are trained only with the converged checkpoints of their partners (FCP_{-T} for "FCP minus time").
2. To test whether FCP would benefit from additional diversity in its partner population, we evaluate an augmentation of FCP in which the population of SP partners varies not just in random seed, but also in architecture (FCP_{+A} for "FCP plus architectural variation").
3. To test whether architectural variation can serve as a full replacement for playing with past checkpoints, we evaluate the combination of both modifications ($\text{FCP}_{-T,+A}$).

2.3 Environment

Following prior work on zero-shot coordination in human-agent interaction, we study the Overcooked environment (see Figure 3) [12, 13, 38, 50, 70]. We draw particular inspiration from the environment in Carroll et al. [12]. For full details, see Appendix A.

In this environment, players are placed into a gridworld kitchen as chefs and tasked with delivering as many cooked dishes of tomato soup as possible within an episode. This involves a series of sequential high-level actions to which both players can contribute: collecting tomatoes, depositing them into cooking pots, letting the tomatoes cook into soup, collecting a dish, getting the soup, and delivering it. Upon a successful delivery, both players are rewarded equally.

To effectively complete the task, players must learn to navigate the kitchen and interact with objects in the correct order, all while maintaining awareness of their partner's behavior to coordinate with them. This environment therefore presents the challenges of both movement and strategic coordination.

Each player observes an egocentric RGB view of the world, and at every step can perform one of six actions: `stand still`, `move {up, down, left, right}`, `interact`. The behavior of `interact` varies based on the cell which the player is facing (e.g. place tomato on counter).

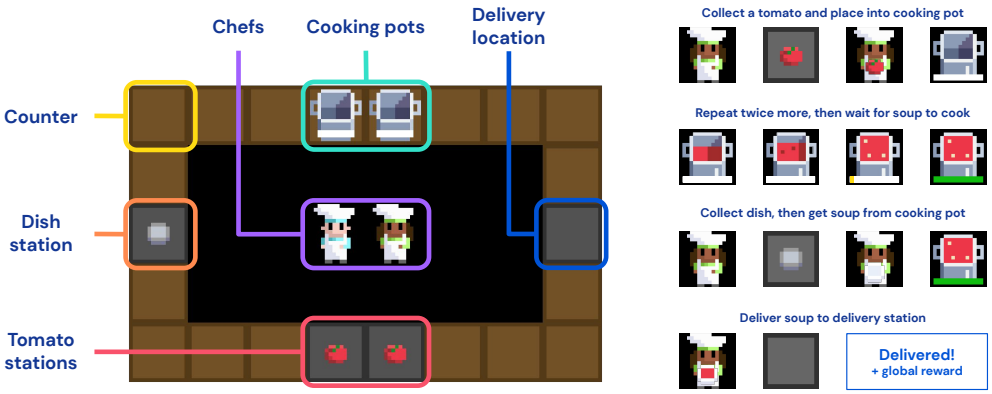


Figure 3: **The Overcooked environment:** a two-player common-payoff game in which players must coordinate to cook and deliver soup.

Figure 4: **Layouts:** the kitchens which agents and humans play in, each emphasizing different coordination strategies. Highlighted in bold are the terms used to refer to each in the rest of this paper.

2.4 Implementation details

Here we highlight several key implementation details for our training methods. For full details, including the architectures, hyperparameters, and compute used, please see Appendix B.

For our reinforcement learning agents, we use the V-MPO [65] algorithm along with a ResNet [26] plus LSTM [29] architecture which we found led to optimal behavior across all layouts. Agents are trained using a distributed set of environments running in parallel [17], each sampling two agents from the training population to play together every episode.

Both PP and FCP are trained with a population size of $N = 32$ agents which are sampled uniformly. For FCP, we use 3 checkpoints for each agent, therefore incurring no additional training burden: (1) at initialization (i.e. a low-skilled agent), (2) at the end of training (i.e. a fully-trained expert agent), and (3) at the middle of training, defined as when the agent reaches 50% of its final reward (i.e. an average-skilled agent). When varying architecture for the training partners of the FCP_{+A} and

$\text{FCP}_{-T,+A}$ variants, we vary whether the partners use memory (i.e. LSTM vs not) and the width of their policy and value networks (i.e. 16 vs 256). In total, we train 8 agents for each of the 4 combinations, leaving the total population size of $N = 32$ unchanged, ensuring a fair comparison.

To train agents via behavioral cloning [58], we use the open-source Acme [30] to learn a policy from human gameplay data. Specifically, we collected 5 human-human trajectories of length 1200 time steps for each of the 5 layouts, resulting in 60k total environment steps. We divide this data in half and train two BC agents: (1) a partner for training a BCP agent, and (2) a "human proxy" partner for agent-agent evaluation. Following Carroll et al. [12], we use a set of feature-based observations for the agents (as opposed to RGB) and generate comparable results: performance is higher on 3 layouts (`asymmetric`, `cramped`, and `ring`) but poorer on the other 2 (`circuit` and `forced`).

3 Related work

Ad-hoc team play There is a large and diverse body of literature on ad-hoc team-play [5, 66], also known as zero-shot coordination [31]. Prior work based in game-theoretic settings has suggested the benefits of planning [71], online learning [51], and novel solution concepts [2], to name a few examples. More recently, multi-agent deep reinforcement learning has provided the tools to scale to more complex gridworld or continuous control settings, leading to work on hierarchical social planning [36], adapting to existing social conventions [40, 62], trajectory diversity [45], and theory of mind [14]. Ad-hoc team-play among novel agent partners is also an object of active study in the emergent communication literature [10, 11, 43]. This prior work has tended to focus on generalization to held-out agent partners as a proxy for human co-players.

Collaborative play with novel humans has been evaluated more actively in the context of training agent assistants; see for instance [57, 68]. To our knowledge, our FCP agents represent the state-of-the-art in coordinating with novel human partners on an equal footing of capabilities in a rich gridworld environment, as measured by the challenge tasks in Carroll et al. [12].

Diversity in multi-agent reinforcement learning In multi-agent reinforcement learning, agents that train with behaviorally diverse populations of game partners tend to demonstrate stronger performance than their self-play counterparts. For example, across a range of multi-agent games, generalization to held-out populations can be improved by training larger and more diverse populations [13, 42, 50]. In mixed-motive settings, cooperation among agents can be encouraged through social diversity, such as in player preferences and rewards [3, 47, 49]. Similarly, competitiveness can be optimized through selective matchmaking between increasingly diverse agents [24, 39, 69].

Despite the increased focus on improving multi-agent performance, evaluation has typically been constrained to agent-agent settings. High-performing agents have infrequently been evaluated with humans, particularly in non-competitive domains [16]. We add to this growing literature, showing that training with diversity is a powerful approach for effective human-agent collaboration.

Human-agent interaction In recent years, increased attention has been directed toward designing machine learning agents capable of collaborating with humans [41, 57, 68, 72] (see also [16] for a broader review on Cooperative AI). Tylkin et al. [68] is particularly notable in also demonstrating that partially trained agents can be useful learning targets for human helpers, although in a different domain (cooperative Atari). Our method, FCP, can be seen as extending theirs by training with multiple "skill levels" and random seeds, rather than just one, which we demonstrate to be crucial to our agents' performance (Tables 1 and 2 and Figure 7b).

A key preceding entry in this research area is Carroll et al. [12], who similarly investigated human-agent coordination in Overcooked. We use their method (BCP) as a baseline throughout our experiments (Section 2.2). Relative to BCP, our approach removes the need for the expensive step of human data collection for agent training. Furthermore, through our novel human-agent experimental design, we go beyond objective performance metrics to compare the subjective preferences that agents generate. For a detailed comparison of methods and results, see Appendix E.

4 Zero-shot coordination with agents

In this section, we evaluate our FCP agent, its ablations, and the baselines with held-out agents.

4.1 Evaluation method: collaborative evaluation with agent partners

Our primary concern in this work is generalization to novel *human* partners (as investigated in Section 5). However, just as collecting human-human data for behavioral cloning is expensive, so too is evaluating agents with humans. Consequently, we instead use generalization to held-out *agent* partners as a cheap proxy of performance with humans. This is then used to guide our model selection process, allowing us to be more targeted with the agents we select for our human-agent evaluations.

We evaluate with three held-out populations:

1. A BC model trained on human data, H_{proxy}, intended as a proxy of generalization to humans, as done by Carroll et al. [12].
2. A set of self-play agents varying in seed, architecture, and training time (specifically, held-out seeds of the $N = 32$ partners trained for the FCP_{+A} agent; see Section 2.4). These are intended to test generalization to a diverse yet still skillful population.
3. Randomly initialized agents intended to test generalization to low-skill partners.

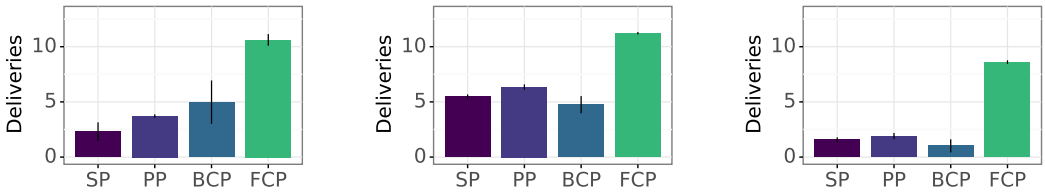
For all results, we report the average number of deliveries made by both players within an episode, aggregated across the 5 different layouts from Figure 4 (with the per-layout results reported in Appendix C.2). We estimate mean and standard deviation across 5 random seeds. For each seed, we evaluate the agent with all members of the held-out population for 10 episodes per agent-partner pair.

4.2 Results

Finding 1: FCP significantly outperforms all baselines

To begin, we compare our FCP agent and the baselines when partnered with the three held-out populations introduced above. As can be seen in Figure 5, FCP significantly outperforms all baselines when partnered with all three held-out populations. Notably, it performs better than BCP with H_{proxy}, even though BCP trains with such a model and FCP does not. Similar to Carroll et al. [12], we find that BCP significantly outscores SP.

When paired with a randomly initialized partner which behaves suboptimally, we see an even greater difference between FCP and the baselines. Given that FCP is trained with non-held-out versions of such agents, it may not be surprising that it does so well with partners that behave poorly. However, what is surprising is how brittle the other training methods are. This suggests that they may not perform well with humans who are not highly skilled players, which we will see in Section 5.



(a) With H_{proxy}. (b) With diverse SP agents. (c) With random agents.

Figure 5: **Agent-agent collaborative evaluation:** Performance of each agent when partnered with each of the held-out populations (Section 4.1) in episodes of length $T = 540$. Importantly, FCP scores higher than all baselines with a variety of test partners. Error bars represent standard deviation over five random training seeds. Plots aggregate data across kitchen layouts; results calculated by individual layout can be found in Appendix C.2.

Finding 2: Training with past checkpoints is the most beneficial variation for performance

Next, we investigate how the different training partner variations influence FCP's performance. In particular, we separately ablate the past checkpoints (T) and architecture (A) variations, evaluating them with the same partners as in Figure 5. The results of this evaluation are presented in Table 1.

Comparing the FCP and FCP_{-T} columns, we see that removing past checkpoints from training significantly reduces performance. Comparing the FCP and FCP_{+A} columns, we see that adding architectural variation to the training population offers no improvement over training with past

Partner	FCP	FCP$_{-T}$	FCP$_{+A}$	FCP$_{-T,+A}$
H_{proxy}	10.6 ± 0.5	4.7 ± 0.4	9.9 ± 0.6	7.0 ± 0.8
Diverse SP	11.2 ± 0.1	6.9 ± 0.1	11.1 ± 0.4	8.6 ± 0.4
Random	8.6 ± 0.2	1.0 ± 0.1	8.4 ± 0.4	3.2 ± 0.5

Table 1: **Ablation results:** Performance of each variation of FCP – training with past partner checkpoints (T for time) and adding partner variation in architecture (A). Scores are mean deliveries with standard deviation over 5 random seeds. Notably, we find that the inclusion of past checkpoints is essential for strong performance (FCP > FCP$_{-T}$), and additionally including architectural variation does not improve performance (FCP $\approx$ FCP$_{+A}$). However, architectural variation is better than no variation, improving performance when past checkpoints are not available (FCP$_{-T,+A}$ > FCP$_{-T}$).

checkpoints. However, comparing the FCP$_{-T}$ and FCP$_{-T,+A}$ columns, we see that without training with past checkpoints, architectural variation in the population does improve performance.

5 Zero-shot coordination with humans

Ultimately, our goal is to develop agents capable of coordinating with novel human partners. In this section, we run an online study to evaluate our FCP agent and the baseline agents in collaborative play with human partners.

Figure 6: **Human-agent collaborative study:** For our human-agent collaboration study, we recruited participants online to play games with FCP and baseline agents. Participants played a randomized sequence of episodes with different agent partners and kitchen layouts. After every two episodes, participants reported the direction and strength of their preference between their last two partners.

5.1 Evaluation method: collaborative evaluation with human participants

To test how effectively FCP's performance generalizes to human partners, we recruited participants from Prolific [18, 55] for an online collaboration study ($N = 114$; 37.7% female, 59.6% male, 1.8% nonbinary; median age between 25–34 years). We used a within-participant design for the study: each participant played with a full cohort of agents (i.e. generated through every training method). This design allowed us to evaluate both objective performance as well as subjective preferences.

Participants first read game instructions and played a short tutorial episode guiding them through the dish preparation sequence (see Appendix D.1.1 for instruction text and study screenshots). Participants then played 20 episodes with a randomized sequence of agent partners and kitchen layouts. Episodes lasted $T = 300$ steps (1 minute) each. After every two episodes, participants reported their preference over the agent partners from those episodes on a five-point Likert-type scale. After playing all 20 episodes, participants completed a debrief questionnaire collecting standard demographic information and open-ended feedback on the study. Our statistical analysis below primarily relies upon the repeated-measures analysis of variance (ANOVA) method. See Appendix D for additional details of our study design and analysis, including independent ethical review.

5.2 Results

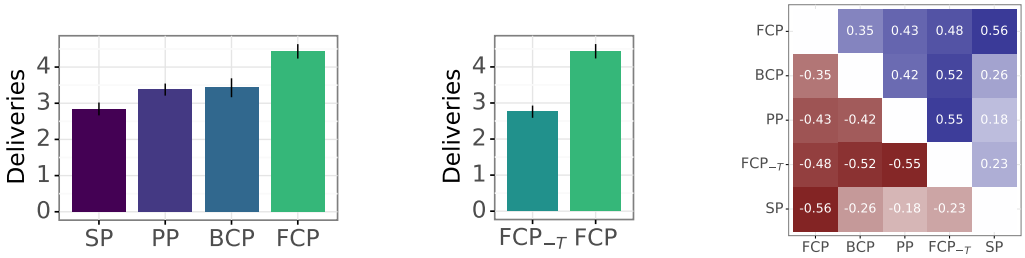
Finding 1: FCP coordinates best with humans, achieving the highest score across maps

To begin, we compare the objective team performance supported by our FCP and baseline agents. The strong FCP performance observed in agent-agent play generalizes to human-agent collaboration:

the FCP-human teams significantly outperform all other agent-human teams, achieving the highest average scores across maps, every $p < 0.001$ (Figure 7a), while performing as well as or better than the other teams on each individual map (see Appendix D.3). Echoing the results from our agent-agent ablation experiments (Table 1), the inclusion of past checkpoints in training proves critical to FCP's strong performance, $p < 0.001$ (Figure 7b). Similar to Carroll et al. [12], we find that BCP outscores SP when collaborating with human players, $p < 0.001$.

Finding 2: Participants prefer FCP over all baselines

FCP's strong collaborative performance carries over to our participants' subjective partner preferences. Participants expressed a significant preference for FCP partners over all other agents, including BCP, with every $p < 0.05$ (Figure 7c). Notably, while human-BCP and human-PP teams did not significantly differ in their completed deliveries, participants reported significantly preferring BCP over PP, $p = 0.003$, highlighting the informativeness of our subjective analysis.



(a) Number of deliveries by partner (FCP and baselines).

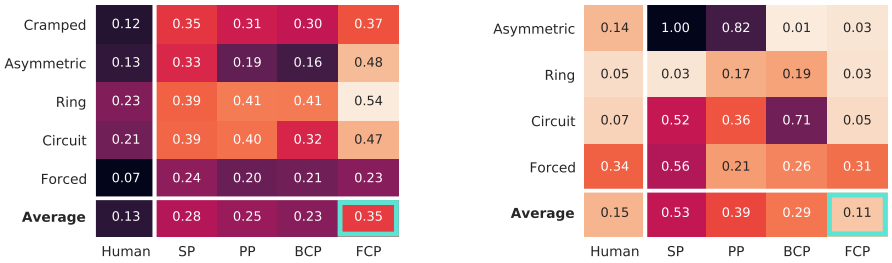
(b) Number of deliveries by partner (FCP and FCP$_{-T}$).

(c) Participant preference for row partner over column partner.

Figure 7: **Human-agent collaborative evaluation**: Evaluation and preference metrics from human-agent play in episodes of length $T = 300$. Error bars represents 95% confidence intervals, calculated over episodes. Plots aggregate data across kitchen layouts; results calculated by individual layout can be found in Appendix D.3.

5.3 Exploratory behavioral analysis

To better understand how the human-agent scores and preferences may have arisen, here we analyze the resulting action trajectories of each human and agent player in our experiment.



(a) Proportion of episode spent moving.

(b) Differences in pot preference.

Figure 8: **Behavioral analysis:** (a) FCP is able to move most frequently (35% of the time), corresponding to the best movement coordination with human partners. (b) FCP exhibits the most equal preferences over cooking pots (0.11 difference), aligning with human preferences. Values are calculated as the absolute difference in preferences between the two pots; 1 indicates that the player only uses one of the two available pots, while 0 indicates that the player uses both pots equally.

Finding 1: FCP exhibits the best movement coordination with humans

First, we investigate how much each player moves in an episode (Figure 8a), where moving in a higher fraction of timesteps may suggest fewer collisions and thus better coordination with a partner. Notably, we observe two results: (1) humans rarely move, a behavior which is out-of-distribution for typical training methods (e.g. SP, PP) but is seen in the training distribution for BCP and FCP.

(2) FCP moves the most on all layouts other than Forced, suggesting it is better at coordinating its movement strategy with its partner. This result was also reported by human participants, for example: "I noticed that some of my partners seemed to know they needed to move around me, while others seemed to get 'stuck' until I moved out of their way" (see Appendix D for more examples).

Finding 2: FCP's preferences over cooking pots aligns best with that of humans

Next, we investigate whether there was a preference for a specific cooking pot in the layouts which included two cooking pots (Figure 8b). To do this, we calculate the difference in the number of times each pot was used by each player, where a high value indicates a strong preference for one pot and a low value indicates more equal preference for the two pots.

As can be seen in the FCP column, our agent typically has the most aligned preferences with that of humans (0.11 for FCP to 0.14 for humans). Behaviorally speaking, this means that our agent prefers one cooking pot over the other 55.5% of the time (i.e. a 0.11 point difference). In contrast, all other agents have a strong preference for a single pot. This is a non-adaptive strategy which generalizes poorly to typical human behavior of using both pots, leading to worse performance.

6 Discussion

Summary In this work, we investigated the challenging problem of zero-shot collaboration with humans without using human data in the training pipeline. To accomplish this, we introduced Fictitious Co-Play (FCP) – a surprisingly simple yet effective method based on creating a diverse set of training partners. We found that FCP agents scored significantly higher than all baselines when partnered with both novel agent and human partners. Furthermore, through a rigorous human-agent experimental design, we also found that humans reported a strong subjective preference to partnering with FCP agents over all baselines.

Limitations and future work Our method currently relies on the manual process of initially training and selecting a diverse set of partners. This is not only time consuming, but also prone to researcher biases that may negatively influence the behavior of the created agents. Additionally, while we found FCP with a partner population size of $N = 32$ sufficient here, for more complex games, FCP may require an unrealistically large partner population size to represent sufficiently diverse strategies. To address these concerns, methods for automatically generating partner diversity for common-payoff games may be important. Possibilities include adaptive population matchmaking as been used in competitive zero-sum games [69], as well as auxiliary objectives that explicitly encourage behavioral diversity [19, 45, 46].

Our method requires a known and fixed reward function. We also focus on one domain in order to compare with prior work which has argued that human-in-the-loop training is necessary. Consequently, the resulting agents are only designed to adaptively collaborate on a single task, and not to infer human preferences in general [1, 33, 59]. Moreover, if a task's reward function is poorly aligned with how humans approach the task, our method may well produce subpar partners, as would any method without access to human data. Thus, additional domains and tasks should be studied to better understand how our method generalizes. Targeted experiments to test specific forms of generalization may be especially helpful in this regard [38], as could approaches that procedurally generate environment layouts requiring diverse solutions [22].

Finally, it may be possible to produce even stronger agent assistants by combining the strengths of FCP (i.e. diversity) and BCP (i.e. human-like play). Indeed, Knott et al. [38] recently demonstrated that modifying BCP to train with *multiple* BC partners produces more robust collaboration with held-out agents, a finding that would be interesting to test with human partners.

Societal impact A challenge for this line of work is ensuring agent behavior is aligned with human values (i.e. the AI value alignment problem [23, 59]). Our method has no guarantees that the resulting policy aligns with the preferences, intentions, or welfare of its potential partners. It likewise does not exclude the possibility that the target being optimized for is harmful (e.g. if the agent's partner expresses preferences or intentions to harm others). This could therefore produce negative societal effects either if training leads to poor alignment or if agents are optimized for harmful metrics.

One potential strategy for mitigating these risks is the use of human preference data [15]. Such data could be used to fine-tune and filter trained agents before deployment, encouraging better alignment with human values. A key question in this line of research is how human preference data should be

aggregated—or selected, in the case of expert preferences—when our aim is to create socially aligned agents (i.e. agents that are sufficiently aligned for everyone). Relatedly, targeted research on human beliefs and perceptions of AI [48], and how they steer human-agent interaction, would help inform agent design for positive societal impact. For instance, developers could incorporate specific priors into agents to reinforce tendencies for fair outcomes [20, 32].

Conclusion We proposed a method which is both effective at collaborating with humans and simple to implement. We also presented a rigorous and general methodology for evaluating with humans and eliciting their preferences. Together, these establish a strong foundation for future research on the important challenge of human-agent collaboration for benefiting society.

Acknowledgements

The authors would like to thank Mary Cassin for creating the game sprite art; Rohin Shah, Thore Graepel, and Iason Gabriel for feedback on the draft; Lucy Campbell-Gillingham, Tina Zhu, and Saffron Huang for support in evaluating agents with humans; and Max Kleiman-Weiner, Natasha Jaques, Marc Lanctot, Mike Bowling, and Dan Roberts for useful discussions.

Funding disclosure

This work was funded solely by DeepMind. The authors declare no competing interests.

References

[1] J. Abramson, A. Ahuja, I. Barr, A. Brussee, F. Carnevale, M. Cassin, R. Chhaparia, S. Clark, B. Damoc, A. Dudzik, et al. Imitating interactive intelligence. *arXiv preprint arXiv:2012.05672*, 2020.

[2] S. V. Albrecht and S. Ramamoorthy. A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2013.

[3] B. Baker. Emergent reciprocity and team formation from randomized uncertain social preferences. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[4] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, et al. The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280:103216, 2020.

[5] S. Barrett, A. Rosenfeld, S. Kraus, and P. Stone. Making friends on the fly: Cooperating with new teammates. *Artificial Intelligence*, 242:132–171, 2017.

[6] A. Bauer, D. Wollherr, and M. Buss. Human–robot collaboration: A survey. *International Journal of Humanoid Robotics*, 5(01):47–66, 2008.

[7] G. W. Brown. Iterative solution of games by fictitious play. *Activity analysis of production and allocation*, 13(1):374–376, 1951.

[8] N. Brown and T. Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

[9] N. Brown and T. Sandholm. Superhuman AI for multiplayer poker. *Science*, 365(6456):885–890, 2019.

[10] K. Bullard, F. Meier, D. Kiela, J. Pineau, and J. Foerster. Exploring zero-shot emergent communication in embodied multi-agent populations. *arXiv preprint arXiv:2010.15896*, 2020.

[11] K. Bullard, D. Kiela, J. Pineau, and J. Foerster. Quasi-equivalence discovery for zero-shot emergent communication. *arXiv preprint arXiv:2103.08067*, 2021.

[12] M. Carroll, R. Shah, M. K. Ho, T. Griffiths, S. Seshia, P. Abbeel, and A. Dragan. On the utility of learning about humans for human-AI coordination. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[13] R. Charakorn, P. Manoonpong, and N. Dilokthanakul. Investigating partner diversification methods in cooperative multi-agent deep reinforcement learning. In *International Conference on Neural Information Processing*, 2020.

[14] R. Choudhury, G. Swamy, D. Hadfield-Menell, and A. D. Dragan. On the utility of model learning in HRI. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2019.

[15] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4302–4310, 2017.

[16] A. Dafoe, E. Hughes, Y. Bachrach, T. Collins, K. R. McKee, J. Z. Leibo, K. Larson, and T. Graepel. Open problems in cooperative AI. *arXiv preprint arXiv:2012.08630*, 2020.

[17] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International Conference on Machine Learning (ICML)*, 2018.

[18] P. Eyal, R. David, G. Andrew, E. Zak, and D. Ekaterina. Data quality of platforms and panels for online behavioral research. *Behavior Research Methods*, pages 1–20, 2021.

[19] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations (ICLR)*, 2019.

[20] E. Fehr and K. M. Schmidt. A theory of fairness, competition, and cooperation. *The Quarterly Journal of Economics*, 114(3):817–868, 1999.

[21] J. Foerster, F. Song, E. Hughes, N. Burch, I. Dunning, S. Whiteson, M. Botvinick, and M. Bowling. Bayesian action decoder for deep multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2019.

[22] M. Fontaine, Y.-C. Hsu, Y. Zhang, B. Tjanaka, and S. Nikolaidis. On the importance of environments in human-robot coordination. In *Robotics: Science and Systems (RSS)*, 2021.

[23] I. Gabriel. Artificial intelligence, values, and alignment. *Minds and Machines*, 30(3):411–437, 2020.

[24] M. Garnelo, W. M. Czarnecki, S. Liu, D. Tirumala, J. Oh, G. Gidel, H. van Hasselt, and D. Balduzzi. Pick your battles: Interaction graphs as population-level objectives for strategic diversity. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2021.

[25] Ghost Town Games. Overcooked, 2016.

[26] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[27] J. Heinrich and D. Silver. Deep reinforcement learning from self-play in imperfect-information games. In *NIPS Deep Reinforcement Learning (DRL) Workshop*, 2016.

[28] J. Heinrich, M. Lanctot, and D. Silver. Fictitious self-play in extensive-form games. In *International Conference on Machine Learning (ICML)*, 2015.

[29] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

[30] M. Hoffman, B. Shahriari, J. Aslanides, G. Barth-Maron, F. Behbahani, T. Norman, A. Abdolmaleki, A. Cassirer, F. Yang, K. Baumli, S. Henderson, A. Novikov, S. G. Colmenarejo, S. Cabi, C. Gulcehre, T. L. Paine, A. Cowie, Z. Wang, B. Piot, and N. de Freitas. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*, 2020.

[31] H. Hu, A. Lerer, A. Peysakhovich, and J. Foerster. "Other-Play" for zero-shot coordination. In *International Conference on Machine Learning (ICML)*, 2020.

[32] E. Hughes, J. Z. Leibo, M. G. Phillips, K. Tuyls, E. A. Duéñez-Guzmán, A. G. Castañeda, I. Dunning, T. Zhu, K. R. McKee, R. Koster, H. Roff, and T. Graepel. Inequity aversion improves cooperation in intertemporal social dilemmas. In *Neural Information Processing Systems (NeurIPS)*, 2018.

[33] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei. Reward learning from human preferences and demonstrations in Atari. In *Neural Information Processing Systems (NeurIPS)*, 2018.

[34] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castaneda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, et al. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

[35] S. Javdani, S. S. Srinivasa, and J. A. Bagnell. Shared autonomy via hindsight optimization. *Robotics Science and Systems*, 2015.

[36] M. Kleiman-Weiner, M. K. Ho, J. L. Austerweil, M. L. Littman, and J. B. Tenenbaum. Coordinate to cooperate or compete: abstract goals and joint intentions in social interaction. In *Conference of the Cognitive Science Society (CogSci)*, 2016.

[37] G. Klien, D. D. Woods, J. M. Bradshaw, R. R. Hoffman, and P. J. Feltovich. Ten challenges for making automation a "team player" in joint human-agent activity. *IEEE Intelligent Systems*, 19 (6):91–95, 2004.

[38] P. Knott, M. Carroll, S. Devlin, K. Ciosek, K. Hofmann, A. Dragan, and R. Shah. Evaluating the robustness of collaborative agents. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2021.

[39] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Pérolat, D. Silver, and T. Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In *Neural Information Processing Systems (NIPS)*, 2017.

[40] A. Lerer and A. Peysakhovich. Learning existing social conventions via observationally augmented self-play. In *AAAI/ACM Conference on Artificial Intelligence, Ethics, and Society (AIES)*, 2019.

[41] E. Lockhart, N. Burch, N. Bard, S. Borgeaud, T. Eccles, L. Smaira, and R. Smith. Human-agent cooperation in bridge bidding. *arXiv preprint arXiv:2011.14124*, 2020.

[42] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Neural Information Processing Systems (NIPS)*, 2017.

[43] R. Lowe, A. Gupta, J. Foerster, D. Kiela, and J. Pineau. Learning to learn to communicate. In *ICML Adaptive & Multitask Learning Workshop*, 2019.

[44] R. Lowe, A. Gupta, J. Foerster, D. Kiela, and J. Pineau. On the interaction between supervision and self-play in emergent communication. In *International Conference on Learning Representations (ICLR)*, 2020.

[45] A. Lupu, H. Hu, and J. Foerster. Trajectory diversity for zero-shot coordination. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2021.

[46] A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson. Maven: Multi-agent variational exploration. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[47] K. R. McKee, I. Gemp, B. McWilliams, E. A. Duèñez-Guzmán, E. Hughes, and J. Z. Leibo. Social diversity and social preferences in mixed-motive reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2020.

[48] K. R. McKee, X. Bai, and S. Fiske. Understanding human impressions of artificial intelligence. *PsyArXiv*, 2021.

[49] K. R. McKee, E. Hughes, T. O. Zhu, M. J. Chadwick, R. Koster, A. G. Castaneda, C. Beattie, T. Graepel, M. Botvinick, and J. Z. Leibo. Deep reinforcement learning models the emergent dynamics of human cooperation. *arXiv preprint arXiv:2103.04982*, 2021.

[50] K. R. McKee, J. Z. Leibo, C. Beattie, and R. Everett. Quantifying environment and population diversity in multi-agent reinforcement learning. *arXiv preprint arXiv:2102.08370*, 2021.

[51] F. Melo and A. Sardinha. Ad hoc teamwork by learning teammates' task. *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2015.

[52] B. Mutlu, A. Terrell, and C.-M. Huang. Coordination mechanisms in human-robot collaboration. In *HRI Collaborative Manipulation Workshop*, 2013.

[53] S. Nikolaidis and J. Shah. Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2013.

[54] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving Rubik's Cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[55] E. Peer, L. Brandimarte, S. Samat, and A. Acquisti. Beyond the Turk: Alternative platforms for crowdsourcing behavioral research. *Journal of Experimental Social Psychology*, 70:153–163, 2017.

[56] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[57] P. M. Pilarski, A. Butcher, M. Johanson, M. M. Botvinick, A. Bolt, and A. S. Parker. Learned human-agent decision-making, communication and joint action in a virtual reality environment. In *Multidisciplinary Conference on Reinforcement Learning and Decision Making (RLDM)*, 2019.

[58] D. A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

[59] S. Russell. *Human Compatible: Artificial Intelligence and the Problem of Control*. Penguin, 2019.

[60] D. Sadigh, N. Landolfi, S. S. Sastry, S. A. Seshia, and A. D. Dragan. Planning for cars that coordinate with people: leveraging effects on human actions for planning and active information gathering over human internal state. *Autonomous Robots*, 42(7):1405–1426, 2018.

[61] N. Schurr, J. Marecki, M. Tambe, and P. Scerri. Towards flexible coordination of human-agent teams. *Multiagent and Grid Systems*, 1(1):3–16, 2005.

[62] A. Shih, A. Sawhney, J. Kondic, S. Ermon, and D. Sadigh. On the critical role of conventions in adaptive human-AI collaboration. In *International Conference on Learning Representations (ICLR)*, 2021.

[63] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[64] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419): 1140–1144, 2018.

[65] H. F. Song, A. Abdolmaleki, J. T. Springenberg, A. Clark, H. Soyer, J. W. Rae, S. Noury, A. Ahuja, S. Liu, D. Tirumala, N. Heess, D. Belov, M. Riedmiller, and M. M. Botvinick. V-MPO: On-policy maximum a posteriori policy optimization for discrete and continuous control. In *International Conference on Learning Representations (ICLR)*, 2020.

[66] P. Stone, G. Kaminka, S. Kraus, and J. Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *AAAI Conference on Artificial Intelligence*, 2010.

[67] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[68] P. Tylkin, G. Radanovic, and D. C. Parkes. Learning robust helpful behaviors in two-player cooperative Atari environments. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2021.

[69] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[70] R. E. Wang, S. A. Wu, J. A. Evans, J. B. Tenenbaum, D. C. Parkes, and M. Kleiman-Weiner. Too many cooks: Coordinating multi-agent collaboration through inverse planning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2020.

[71] F. Wu, S. Zilberstein, and X. Chen. Online planning for ad hoc autonomous agent teams. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[72] S. Zheng, A. Trott, S. Srinivasa, N. Naik, M. Gruesbeck, D. C. Parkes, and R. Socher. The AI economist: Improving equality and productivity with AI-driven tax policies. *arXiv preprint arXiv:2004.13332*, 2020.

A Environment details

A.1 Gameplay

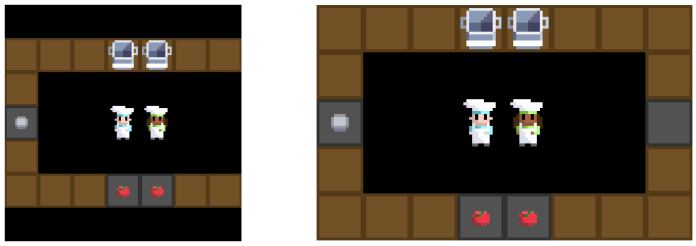
Players are placed in a gridworld environment containing cooking pots, tomato stations, dish stations, delivery locations, and empty counters. Players can move around and interact with these objects. By sequencing certain object interactions, players can pick up and deposit items. Each player (and each counter) can only hold one item at a time.

The objective of each episode is for the players to deliver as many tomato soup dishes as possible to delivery locations. In order to create a tomato soup dish, players must pickup tomatoes and deposit them into the cooking pot. Once there are three tomatoes in the cooking pot, it begins to cook for 20 steps. After 20 steps, the soup is fully cooked. Cooking progress for the soup is tracked by a loading bar overlaying the cooking pot. The loading bar increments over the cooking time and then turns green when the soup is ready for collection.

When the tomato soup is ready for collection, a player holding an empty dish can interact with the cooking pot to pick up the soup. The player can then deliver the soup by interacting with a delivery station while holding the completed dish. A successful delivery rewards both players and removes the dish from the game.

A.2 Observations

Importantly, our agent and human players observe different views of the environment. We found that agents trained better and were more robust using egocentric observations (Figure 9a). However, humans players found this disorienting and so observed the world from a static top-down perspective of the environment (Figure 9b).



(a) Agent observation (cyan). (b) Human observation.

Figure 9: Example observations: (a) Agent players observe a $56 \times 56 \times 3$ egocentric view of the environment (i.e., 7×7 cells with a $8 \times 8 \times 3$ sprite in each cell). (b) Human players observe the full layout from a top-down perspective.

A.3 Actions

Players can take one of the following eight actions each step:

1. `No-op`: The player stays in the same position.
2. `Move up`: Moves the player up one cell.
3. `Move down`: Moves the player down one cell.
4. `Move left`: Moves the player left one cell.
5. `Move right`: Moves the player right one cell.
6. `Interact`: The player interacts with the cell that they are facing.

The outcome of the `Interact` action depends on the current item held by the player (none, empty dish, tomato, or soup), as well as the type of object which they are facing (counter, cooking pot, tomato station, dish station, or delivery station). Depending on these two conditions, the player will either deposit the held item to the object or pick up an item from the object.

A.4 Rewards

Players receive a shared $+20$ reward for every soup they deliver through the delivery station. As a result, they are incentivized to deliver as many soups as possible within an episode. To help scaffold learning, players also receive $+1$ reward each time they deposit a tomato into the cooking pot.

A.5 Layouts

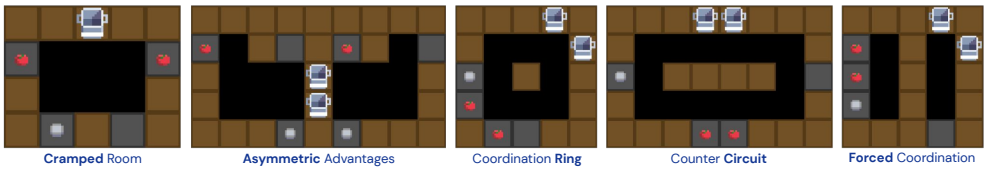


Figure 10: **Layouts:** the kitchens which agents and humans play in, each emphasizing different coordination strategies. Highlighted in bold are the terms used to refer to each in the paper.

- **Cramped:** A tight layout requiring significant movement coordination between the players in order to avoid being blocked by each other.
- **Asymmetric:** A two-room layout with an agent in each. In the left room, the tomato station is far away from the cooking pots while the delivery location is close. In the right room, the tomato station is next to the cooking pots while the delivery station is far. This presents an asymmetric advantage of responsibilities for optimally creating and delivering soups.
- **Ring:** A layout with two equally successful movement strategies – (1) both players moving clockwise, and (2) both players moving anti-clockwise. If players do not coordinate, they will block each other's movement.
- **Circuit:** Players are able to cook and deliver soups by themselves through walking around the entire circuit. However, there exists a more optimal coordinated strategy whereby players pass tomatoes across the counter. Additionally, there are the clockwise and anti-clockwise strategies as in the Ring layout.
- **Forced:** One player is in the left room and second player is in the right room. Consequently, both players are forced to work together in order to cook and deliver soup. The player in the left room can only pass tomatoes and dishes, while the player on the right can only cook the soup and deliver it (using the items provided by the first player).

B Agent details

B.1 Fictitious Co-Play

We provide pseudocode for Fictitious Co-Play (FCP) in Algorithm 1. Unless otherwise stated, all FCP results are with a partner population size of $N = 32$. We used a checkpoint frequency n_c of 1×10^7 steps, resulting in a total of 100 checkpoints saved per training run of 1×10^9 steps. After the first stage of training partners, we filter checkpoints (i.e. F) down to three for each partner: the first checkpoint (i.e. a randomly initialized agent), the last checkpoint (i.e. a fully trained agent),

and the remaining checkpoint that achieves closest to half of the reward of the last checkpoint (i.e. a half-trained agent).

Algorithm 1: Fictitious Co-Play (FCP)

Input: Number of partners N, checkpoint frequency n_c, checkpoint filter F
`// Stage 1: train diverse partner population`
partners = []
for $i = 1$ **to** N **do**
  Initialize agent i.
  $n = 0$ `// step count`
  while *not converged* **do**
    Update agent i in self-play.
    $n \mathrel{+}= 1$
    if $n \mod n_c = 0$ **then**
      Add frozen agent i checkpoint to partners.
`// Stage 2: train FCP agent`
Filter partners with F.
Initialize FCP agent.
while *not converged* **do**
  Sample partner from partners.
  Update FCP in co-play with partner.

B.2 Training settings

Agents are trained using a distributed set of environments running in parallel. Each agent is trained using one GPU on $N \times 200$ environments, where N is the number of agents being trained in the population. Agents are trained for 1×10^9 environment steps which takes between three and eight days depending on the size of the training population. As the environment involves two players, each one samples with replacement from the training population of agents every episode.

B.3 Deep reinforcement learning (DRL)

B.3.1 Architecture and hyperparameters

We used the following architecture for V-MPO [65]. The agent's visual observations were first processed through a 3-section ResNet used in McKee et al. [50]. Each section consisted of a convolution and 3×3 max-pooling operation (stride 2), followed by residual blocks of size 2 (i.e., a convolution followed by a ReLU nonlinearity, repeated twice, and a skip connection from the input residual block input to the output). The entire stack was passed through one more ReLU nonlinearity. All convolutions had a kernel size of 3 and a stride of 1. The number of channels in each section was (16, 32, 32).

The resulting output was then concatenated with the previous action and reward of the agent, and processed by a single-layer MLP with 256 hidden units. This was followed by a single-layer LSTM with 256 hidden units (unrolled for 100 steps), and then a separate single-layer MLP with 256 hidden units to produce the action distribution. For the critic, we used a single-layer MLP with 256 hidden units followed by PopArt normalization (which leads to stronger multi-agent performance [50]).

To train the agent, we used a discount factor of 0.99, batch sizes of 16, and the Adam optimizer (learning rate of 0.0001). We configured V-MPO with a target network update period of 100, $k = 0.5$, and an epsilon temperature of 0.1. For PopArt normalization, we used a scale lower bound of $1\mathrm{e}{-2}$, an upper bound of $1\mathrm{e}6$, and a learning rate of $1\mathrm{e}{-3}$.

B.3.2 Training curves

B.4 Behavioral cloning (BC)

B.4.1 Architecture and hyperparameters

We trained a single network consisting of five sub-networks (1 per layout), each being a three-layer MLP with 256 hidden units per layer. The network was then trained using all of the recorded human

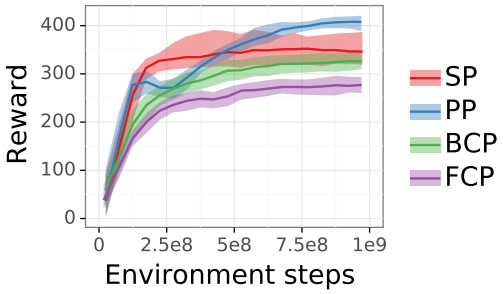


Figure 11: Training curves for agents evaluated. Median and 75% confidence interval over 5 seeds. Note that agents vary in the partners they train with, so the scores should not be directly compared to each other (e.g. FCP training partners include randomly initialized agents).

trajectories (i.e. on all layouts), with the sub-network conditioned on the trajectory's layout feature. We used a batch size of 256 and and the Adam optimizer (learning rate = 0.0003).

B.4.2 Feature-based observations

As learning from RGB observations requires a significant amount of human data, we instead opted to handcraft a set of features for our BC agent to learn from. In particular, we used the following features: player position and orientation, current held item (as a one-hot vector), the relative position of the other player, the state of each cooking pot (as a one-hot vector of empty, 1 tomato, 2 tomato, 3 tomatoes, and cooked), faced cell is empty, each adjacent cell is empty (length 4), and the relative distance to each object (tomato, dish, soup, and delivery location). For our behavioral cloning, we also included the layout name.

B.4.3 Training details

We collected 5 human-human trajectories of length 1200 timesteps for each of the 5 layouts, resulting in 60,000 total environment steps. We then divided this data in half to train two BC agents: (1) a partner for training a BCP agent (H_{partner}), and (2) a "human proxy" partner for agent-agent evaluation (H_{proxy}).

H_{partner} was trained for 32,000 gradient steps (273 epochs) and H_{proxy} was trained for 61,000 gradient steps (520 epochs). Training time was based on peak reward in self-play, analogous to early stopping.

C Zero-shot coordination with agents

C.1 Evaluation details

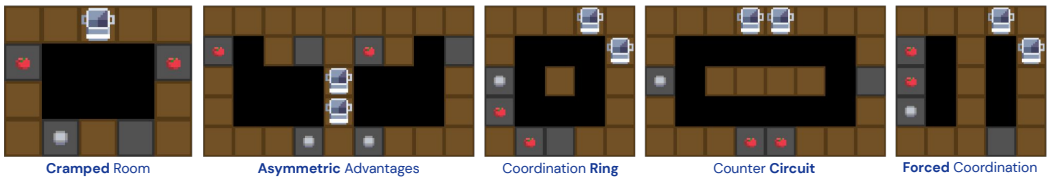
For each of SP, BCP, and FCP, we trained 5 random seeds per agent. For PP, we trained a single 32-agent population and chose the first 5 agents for evaluation. All results are averaged across these seeds.

For the held-out evaluation populations, the "diverse SP" group consisted of 60 self-play agents varying in random seed (5 seeds), architecture (4 variants: LSTM vs feedforward, policy and value network widths of 16 vs 256), and training time (3 variants: randomly initialized, half-trained, and fully-trained). The architectural variation is the same type used for the FCP_{+A} agent in Table 1, while the training time variation is the same used for producing partners for FCP. The random agents population were a subset of the diverse SP group, consisting of the 5 seeds of randomly initialized agents. Finally, H_{proxy} was the BC agent trained on the held-out half of human-human trajectories.

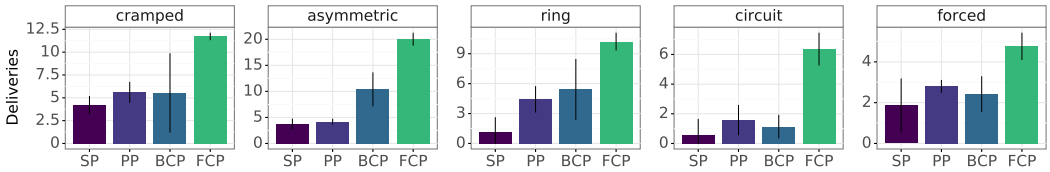
For each pair of agents consisting of one random seed of an agent to be evaluated and one random seed of an agent from a target evaluation population, we played 10 games of length $T = 540$ steps for each layout.

C.2 Additional results

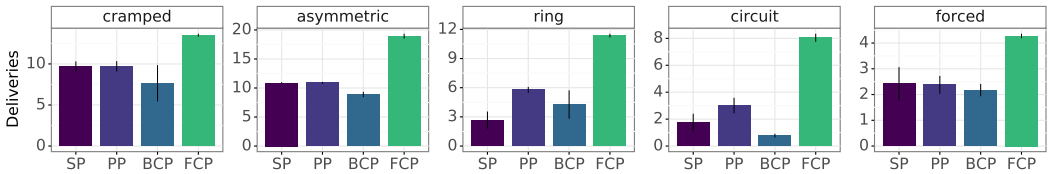
Per-layout results



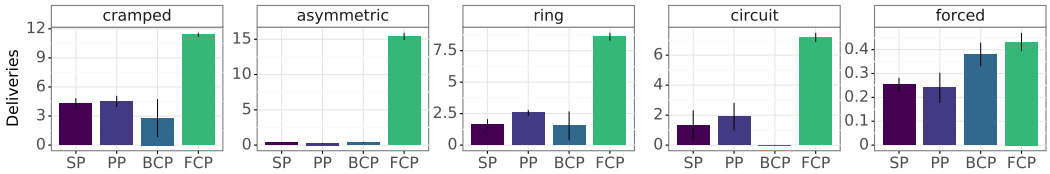
(a) Layouts for reference.



(b) With H_{proxy}.



(c) With diverse SP agents.



(d) With random agents.

Figure 12: **Agent-agent collaborative evaluation, per-layout:** Performance of each agent when partnered with each of the held-out populations in episodes of length $T = 540$. Error bars represent standard deviation over five random training seeds. FCP outperforms all baselines on every map with every partner population.

Influence of population size on performance

As we create a training population to train our agent, a natural question to ask is how the size of that population influences the performance of the agent. In Table 2, we present the results of varying N from 4 to 128.

N	Deliveries
4	8.4 (0.3)
8	9.1 (0.4)
16	10.2 (0.4)
32	10.6 (0.5)
64	10.4 (0.3)
128	10.8 (0.6)

Table 2: Performance of FCP with H_{proxy}, as a function of number of training partners (N): larger populations lead to stronger agents and more deliveries. The right column presents mean deliveries with standard deviation over 5 random seeds in parentheses.

We observe a consistent increase in performance as N increases, plateauing around $N = 32$ training partners. This supports the findings of prior work that larger populations are typically stronger, to a point [38, 42, 50]. Consequently, we selected $N = 32$ as our population size across all experiments.

D Zero-shot coordination with humans

D.1 Experimental design

To test how effectively the FCP and baseline agents collaborate with humans in a zero-shot setting, we recruited $N = 114$ participants from Prolific, an online participant recruitment platform [18, 55]. Inclusion criteria were residence in the United States, a minimum approval rate of 95% on prior Prolific studies, and a minimum of 20 prior approved studies.

The study consisted of multiple tutorial pages explaining the game rules and dynamics (Figures 13 and 14), a single-player practice episode (Figure 15), multiple two-player episodes with agent partners (Figures 16-18), and a debrief questionnaire collecting open-ended feedback and demographic information. Each participant played multiple two-player episodes on different layouts. By the end of the study, each participant had collaborated with agents generated through every training method (i.e., the study had a within-participant design). We incentivized game performance: participants earned \$0.10 for each dish served, resulting in a cumulative performance bonus at the end of the study. Study sessions lasted 31.2 minutes on average, with a compensation base of \$4.00 and an average bonus of \$6.74.

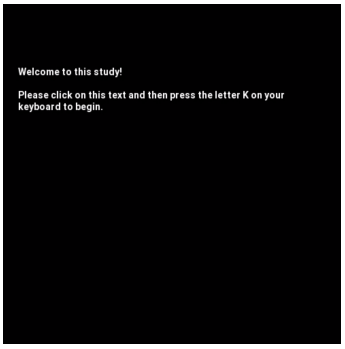
DeepMind's independent research ethics committee conducted ethical review for the project and offered a favorable opinion for the study protocol (#19/001), indicating that it presented minimal risk to participants. All participants provided informed consent for the study.

Each participant played with a randomized sequence of agent partners and game layouts. Specifically, we first randomized the order of the five layouts. Layouts were repeated four times in the sequence, so that each participant played 20 episodes total (four episodes on each layout). To generate the sequence of agents that plays in the four episodes for each layout, we sampled four agents without replacement.

D.1.1 Screenshots

Here we include screenshots of our human-agent collaborative study:

1. Instruction and tutorial screens (Figures 13 and 14).
2. Playing a solo practice episode (Figure 15).
3. Instructions on the episodes with a partner (Figure 16).
4. Playing episode 1 with Partner A (Figure 17).
5. Playing episode 2 with Partner B (Figure 18).
6. Preference elicitation between Partners A and B (Figure 19).
7. Repeat steps 4-6 for 18 more episodes.



Screen 1: Welcome participants to the experiment.

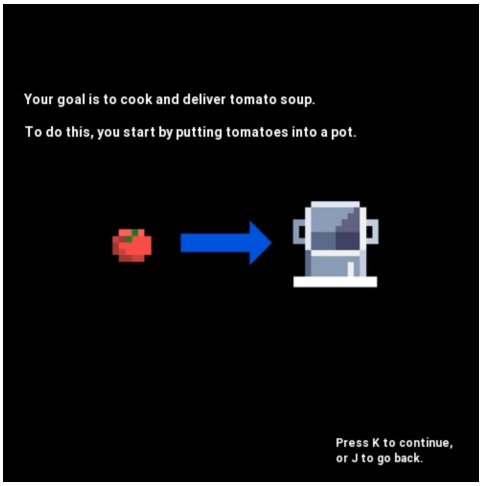
Screen 2: Explain the keyboard controls.

Screen 3: Provide overview of the experiment and bonus.

Screen 4: Introduce the participant's controllable chef.

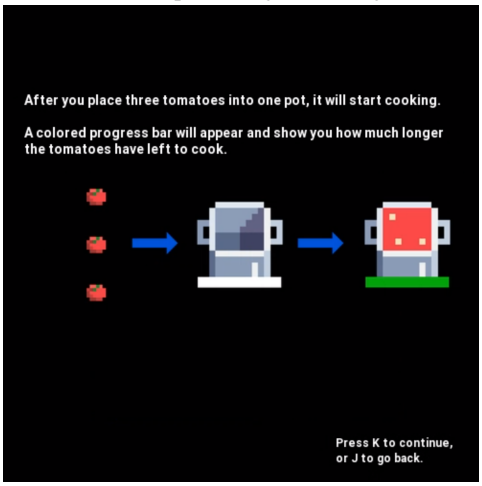
Screen 5: Introduce a generic kitchen layout.

Figure 13: Screenshots of tutorial and instruction screens.

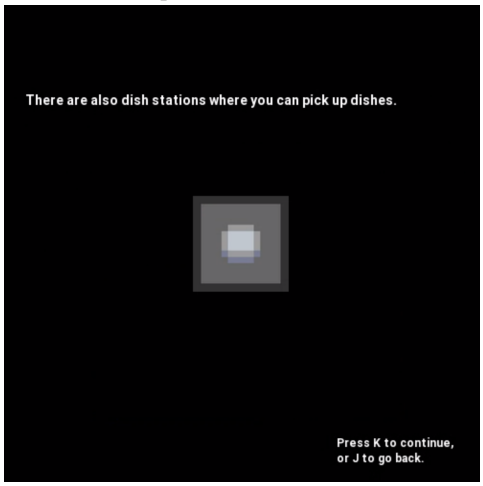


Screen 6: Explain the goal of the game.

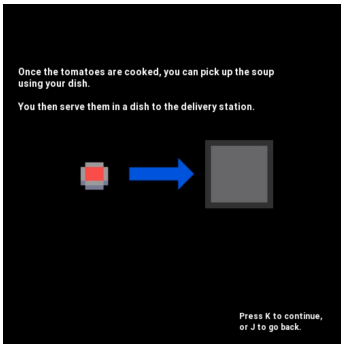
Screen 7: Explain how to collect tomatoes.



Screen 8: Explain how to cook soup.



Screen 9: Explain how to collect dishes.



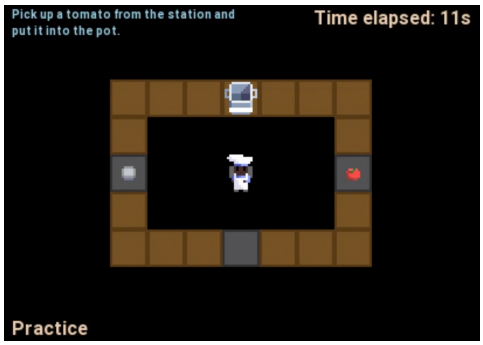
Screen 10: Explain how to deliver soup.

Screen 11: Explain the movement controls for the chef.

Screen 12: Explain the chef's `interact` action.

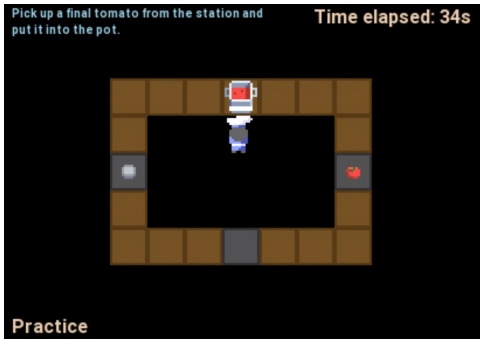
Figure 14: Screenshots of tutorial and instruction screens.

(a) Screen 13: Introduce the practice episode.

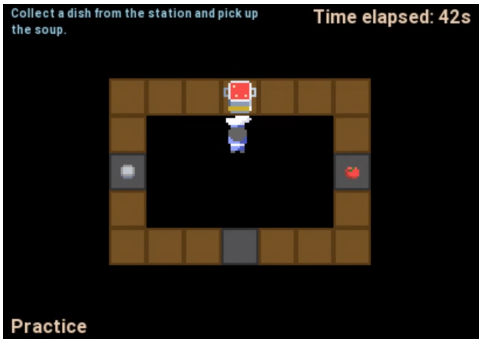


(b) Screen 14: First part of the practice episode.

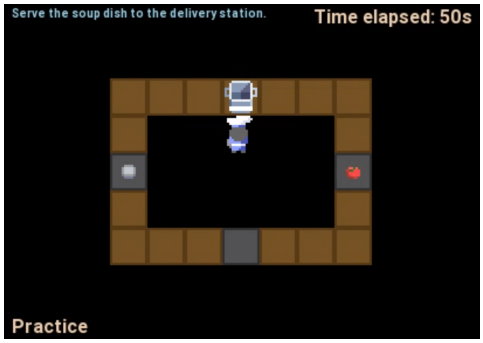
(c) Screen 14: Second part of the practice episode, after the participant has placed one tomato into the cooking pot.



(d) Screen 14: Third part of the practice episode, after the participant has placed two tomatoes into the cooking pot.

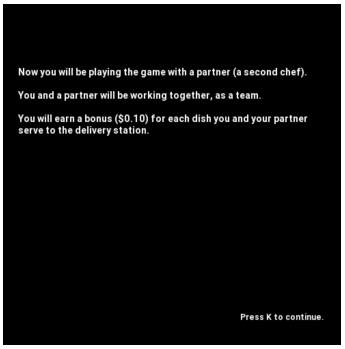


(e) Screen 14: Fourth part of the practice episode, after the participant has placed three tomatoes into the cooking pot.



(f) Screen 14: Final part of the practice episode, after the participant has collected the soup from the cooking pot.

Figure 15: Screenshots of the practice episode.

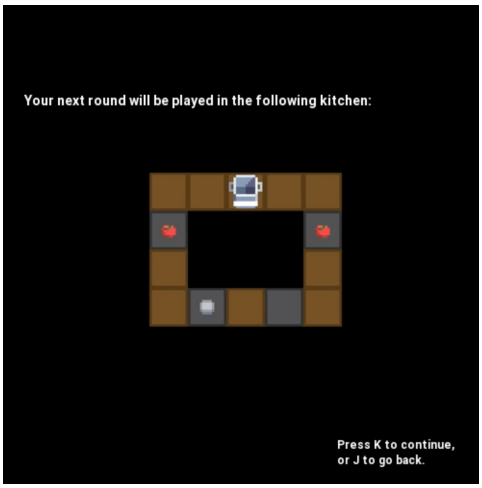


(a) Screen 15: Explain the payment structure for the experiment.

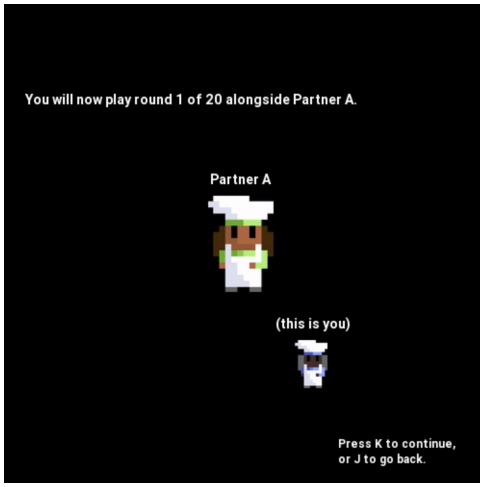
(b) Screen 16: Introduce the participants' partners.

(c) Screen 17: Provide overview of upcoming episodes.

Figure 16: Screenshots of instruction pages.



Screen 18: Introduce kitchen layout for first episode.

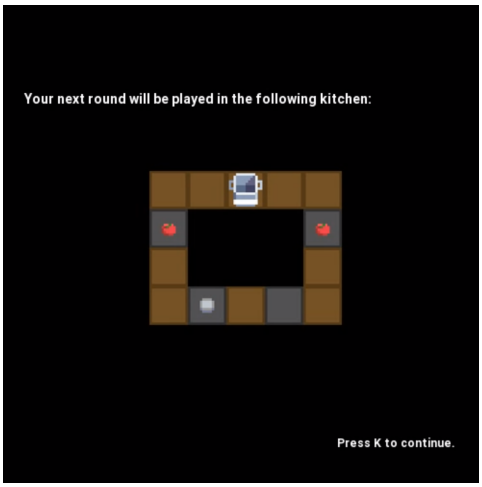


Screen 19: Introduce partner for first episode.

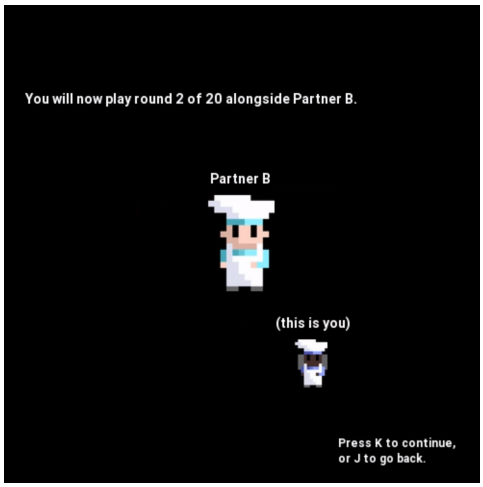
Screen 20: First episode.

Screen 21: Confirm completion of first episode.

Figure 17: Screenshots of first episode.



(a) Screen 22: Introduce kitchen layout for second episode.

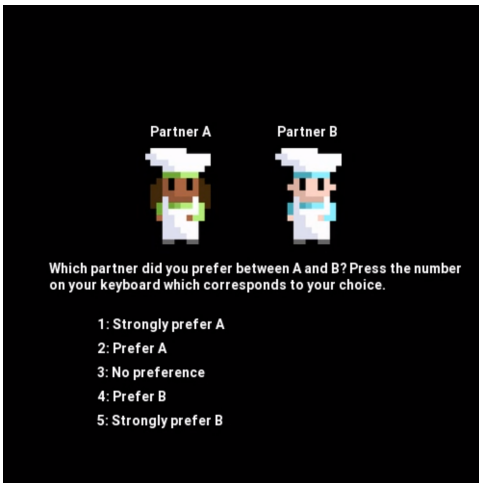


(b) Screen 23: Introduce partner for second episode.

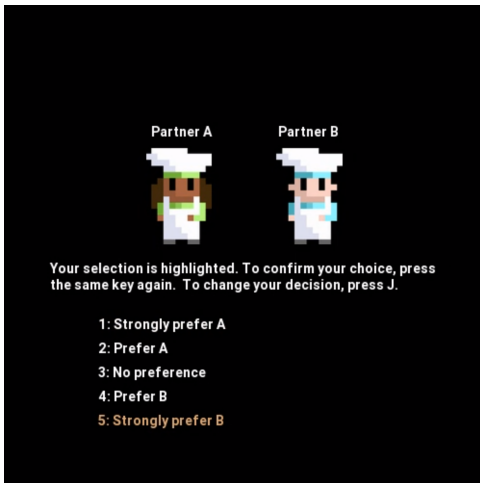
(c) Screen 24: Second episode.

(d) Screen 25: Confirm completion of second episode.

Figure 18: Screenshots of second episode.



(a) Screen 26: Elicit participant's preference between partners.



(b) Screen 27: Confirm participant's preference between partners.

Figure 19: Screenshots of preference elicitation over first and second episodes.

D.2 Analytic details

We run a repeated-measures analysis of variance (ANOVA) to compare average team deliveries for each agent partner, with a random effect incorporated for each participant. Pairwise contrasts (using the Tukey method to adjust for multiple comparisons) indicated that the FCP agent achieved significantly higher delivery totals relative to the BCP agent, $t(1707) = 7.8$ ($p < 0.001$), the PP agent, $t(1707) = 8.2$ ($p < 0.001$), and the SP agent $t(1707) = 12.3$ ($p < 0.001$). The BCP agent also scored significantly higher than the SP agent, $t(1707) = 4.5$ ($p < 0.001$).

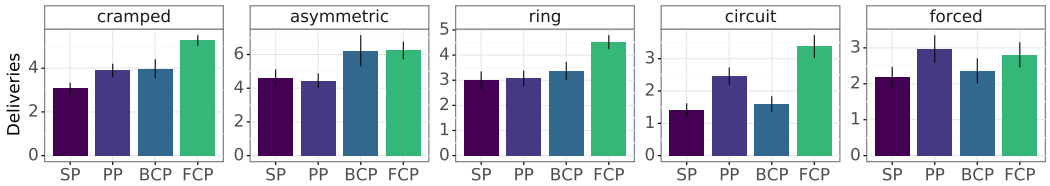
We conduct the same analysis to compare FCP against the FCP$_{-T}$ ablation. Human-agent teams involving the FCP agent completed significantly more deliveries than did those with the FCP$_{-T}$ agent, $t(797) = 14.4$ ($p < 0.001$).

To test whether participants preferred the FCP agent over other agent partners, we similarly fit a repeated-measures ANOVA on preferences elicited between FCP and any other agent, including the identity of the other agent as the sole predictor variable, as well as a random effect for each participant. Participants expressed significantly greater preferences for FCP over the SP agent, $t(329.6) = 4.9$ ($p < 0.001$), the PP agent, $t(399.0) = 2.5$ ($p = 0.011$), the BCP agent, $t(343.1) = 2.0$ ($p = 0.047$), and the FCP$_{-T}$ agent, $t(384.7) = 3.0$ ($p = 0.003$).

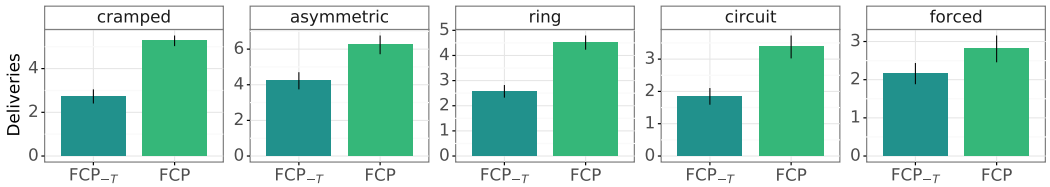
Finally, we run a repeated-measures ANOVA (again with a random effect for participants) to understand participants' preferences between the BCP agent and the PP agent. Participants reported significantly favoring the BCP agent over the PP agent, $t(77.8) = 3.1$ ($p = 0.003$).

D.3 Additional quantitative results

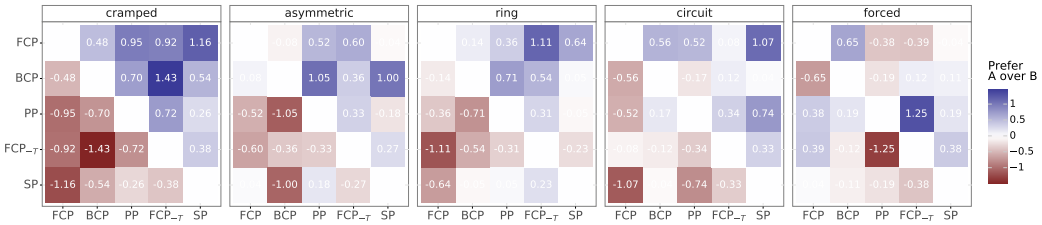
(a) Layouts for reference.



(b) Human-agent team deliveries by map (FCP vs baselines).



(c) Human-agent team deliveries by map (FCP vs FCP$_{-T}$).



(d) Human preferences for agent partners by map.

Figure 20: **Human-agent collaborative evaluation, per-layout:** Performance of each agent when partnered with human participants, as assessed by both objective and subjective metrics. In (a) and (b), error bars represent 95% confidence intervals. In terms of deliveries, FCP performs comparably or better than all baselines on every map. FCP is also the most consistently preferred agent across maps, though on some specific maps (asymmetric and forced) participants occasionally report preferring other agents over FCP.

D.4 Additional qualitative results

At the end of the study, we prompted participants with an open-ended question for feedback on their partners. Many responses touched on adaptability, specialization, and goal compatibility. We include a sample of responses below.

- "Some did not adapt well to what I was doing or they were chaotic and I could not find a way to adapt to them."
- "Sometimes, depending on the layout of the kitchen, my partner seemed to chose a role and stick with it, allowing me to take the other role, and this seemed to be the most efficient way to work. For example, if my partner was 'in charge' of filling the pot, and I was in charge of putting it on a plate and delivering it. Also, I noticed that some of my partners seemed to know they needed to move around me, while others seemed to get 'stuck' until I moved out of their way."

- "I prefer partners which were willing not just to work independently but to let me hand them tomatoes/plates (since I maneuvered slower than them). It was difficult when they did not work on the same goal together but acted independently."
- "The responsive and compatibility of the other partners was really interesting as it took some quite some time to figure out what strategy works best for us and the strategy was almost instant with other partners."
- "Oh man I never laughed so much in my life. My partners were all fun to play with. Some were a bit faster than others. I think the one in pink was the fastest overall."

E Related work

Our work is similar to that conducted by Carroll et al. [12]. Here we provide a summary of the notable differences in experimental design that may contribute to differences in our results.

First, we did not use population-based training (PBT) with $N = 3$ as a baseline. We instead used population-play (PP) with $N = 32$, which shares some similarities with PBT. In our experiments, PP performed significantly better than SP. One potential reason for the performance difference between these two experiments is the small $N = 3$ population size used for PBT; this may have led to a collapse of diversity. This echos a key finding from Jaderberg et al.'s [74] methodological evaluation of PBT: "If the population size is too small (10 or below) we tend to encounter higher variance and can suffer from poorer results". This issue may have been exacerbated by the two-player common-payoff nature of our task.

Second, we trained our agents on an egocentric observation of the environment, as opposed to a top-down layer representation of the whole environment. Egocentric observations can improve agent generalization [75]. The fixed observation size in our approach additionally allowed us to train a single agent to play on all layouts, in contrast with Carroll et al.'s approach of training one agent per layout. We also used the VMPO learning algorithm [65] rather than PPO.

We implemented the game environment in the open-source environment engine DMLab2D [73]. Our implementation followed the implementation reported in the Carroll et al. [12], but was likely not a perfect recreation. We implemented our own human-agent interaction pipeline on top of the DMLab2D environment, with additional instructions and text to explain the game (as described in Section D). In our human-agent experiments, episodes lasted 300 steps over 60 seconds (5 FPS); in Carroll et al. [12], episodes lasted 400 steps over 60 seconds (6.66 FPS).

Finally, our BC agents were trained on 12,000 environment steps per layout, rather than 18,000 steps per layout, due to data collection limitations. We used an architecture with larger layers and an overall higher number of layers than used in Carroll et al. [12]. The observable features used to train the BC agent differed slightly; nonetheless, we observed no noticeable difference in performance. With our BC approach, we found that the random-action heuristic Carroll et al. [12] used was unnecessary.

References

[73] C. Beattie, T. Köppe, E. A. Duéñez-Guzmán, and J. Z. Leibo. DeepMind Lab2D. *arXiv preprint arXiv:2011.07027*, 2020.

[74] M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

[75] C. Ye, A. Khalifa, P. Bontrager, and J. Togelius. Rotation, translation, and cropping for zero-shot generalization. In *IEEE Conference on Games (CoG)*, 2020.