A Cordial Sync: Going Beyond Marginal Policies for Multi-Agent Embodied Tasks Supplementary Material

Unnat Jain[1⋆], Luca Weihs[2⋆], Eric Kolve[2], Ali Farhadi[3], Svetlana Lazebnik[1], Aniruddha Kembhavi[2,3], and Alexander Schwing[1]

[1] University of Illinois at Urbana-Champaign
[2] Allen Institute for AI [3] University of Washington

A Supplementary Material

This supplementary material provides:

A.1 The conditions for a collection of actions to be considered *coordinated*.
A.2 An example showing that standard independent multi-agent action sampling makes it impossible to, even in principle, obtain an optimal joint policy.
A.3 Training details including hyperparameter choices, hardware configurations, and reward structure. We also discuss our upgrades to AI2-THOR.
A.4 Additional discussion, tables, and plots regarding our quantitative results.
A.5 Additional discussion, tables, and plots of our qualitative results including a description of our supplementary video (https://youtu.be/I_Evs5Bol6k) as well as an in-depth quantitative evaluation of communication learned by our agents.

A.1 Action restrictions

We now comprehensively describe the restrictions defining when actions taken by agents are globally consistent with one another. In the following we will, for readability, focus on the two agent setting. All conditions defined here easily generalize to any number of agents. Recall the sets $\mathcal{A}^{\text{NAV}}, \mathcal{A}^{\text{MWO}}, \mathcal{A}^{\text{MO}}$, and $\mathcal{A}^{\text{RO}}$ defined in Sec. 3. We call these sets the *modalities of action*. Two actions $a^1, a^2 \in \mathcal{A}$ are said to be of the same modality if they both are an element of the same modality of action. Let a^1 and a^2 be the actions chosen by the two agents. Below we describe the conditions when a^1 and a^2 are considered *coordinated*. If the agents' actions are uncoordinated, both actions fail and no action is taken for time t. These conditions are summarized in Fig. 2a.

Same action modality. A first necessary, but not sufficient, condition for successful coordination is that the agents agree on the modality of action to perform. Namely, both a^1 and a^2 are of the same action modality. Notice the block diagonal structure in Fig. 2a.

⋆ denotes equal contribution by UJ and LW

No independent movement. Our second condition models the intuitive expectation that if one agent wishes to reposition itself by performing a single-agent navigational action, the other agent must remain stationary. Thus, if $a^1, a^2 \in \mathcal{A}^{\text{NAV}}$, then (a^1, a^2) are coordinated if and only if one of a^1 or a^2 is a PASS action. The $\{1, 2, 3, 4\}^2$ entries of the matrix in Fig. 2a show coordinated pairs of single-agent navigational actions.

Orientation synchronized object movement. Suppose that both agents wish to move (with) the object in a direction so that $a^1, a^2 \in \mathcal{A}^{\text{MWO}}$ or $a^1, a^2 \in \mathcal{A}^{\text{MO}}$. As actions are taken from an egocentric perspective, it is possible, for example, that moving ahead from one agent's perspective is the same as moving left from the other's. This condition requires that the direction specified by both of the agents is consistent globally. Hence a^1, a^2 are coordinated if and only if the direction specified by both actions is the same in a global reference frame. For example, if both agents are facing the same direction this condition requires that $a^1 = a^2$ while if the second agent is rotated 90 degrees clockwise from the first agent then $a^1 =$ MOVEOBJECTAHEAD will be coordinated if and only if $a^2 =$ MOVEOBJECTLEFT. See the multicolored 4×4 blocks in Fig. 2a.

Simultaneous object rotation. For the lifted object to be rotated, both agents must rotate it in the same direction in a global reference frame. As we only allow the agents to rotate the object in a single direction (clockwise) this means that $a^1 =$ ROTATEOBJECTRIGHT requires $a^2 = a^1$. See the (9, 9) entry of the matrix in Fig. 2a.

While a pair of uncoordinated actions are always unsuccessful, it need not be true that a pair of coordinated actions is successful. A pair of coordinated actions will be unsuccessful in two cases: performing the action pair would result in (a) an agent, or the lifted object, colliding with one another or another object in the scene; or (b) an agent moving to a position more than 0.76m from the lifted object. Here (a) enforces the physical constraints of the environment while (b) makes the intuitive requirement that an agent has a finite reach and cannot lift an object when being far away.

A.2 Challenge 1 (rank-one joint policies) example

We now illustrate how requiring two agents to independently sample actions from marginal policies can result in failing to capture an optimal, high-rank, joint policy.

Consider two agents A^1 and A^2 who must work together to play rock-paper-scissors (RPS) against some adversary E. In particular, our game takes place in a single timestep where each agent A^i, after perhaps communicating with the other agent, must choose some action $a^i \in \mathcal{A} = \{R, P, S\}$. During this time the adversary also chooses some action $a^E \in \mathcal{A}$. Now, in our game, the pair of agents A^1, A^2 lose if they choose different actions (*i.e.*, $a^1 \neq a^2$), tie with the adversary if all players choose the same action (*i.e.*, $a^1 = a^2 = a^E$), and finally win or lose if they jointly choose an action that beats or losses against the adversary's choice following the normal rules of RPS (*i.e.*, win if $(a^1, a^2, a^E) \in \{(R, R, S), (P, P, R), (S, S, P)\}$, lose if $(a^1, a^2, a^E) \in \{(S, S, R), (R, R, P), (P, P, S)\}$).

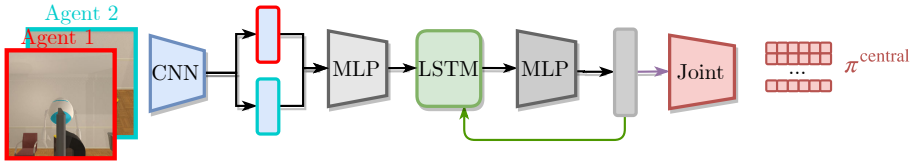

Fig. 6: **Central model architecture.** The central backbone observes the aggregate of all agents' observations. Moreover, the actor in the central model explicitly captures the joint policy distribution.

Moreover, we consider the challenging setting where A^1, A^2 communicate in the open so that the adversary can view their joint policy Π before choosing the action it wishes to take. Notice that we've dropped the t subscript on Π as there is only a single timestep. Finally, we treat this game as zero-sum so that our agents obtain a reward of 1 for victory, 0 for a tie, and -1 for a loss. We refer to the optimal joint policy as Π^*. If the agents operate in a decentralized manner using their own (single) marginal policies, their effective rank-one joint policy equals $\Pi = \pi^1 \otimes \pi^2$.

Optimal joint policy: It is well-known, and easy to show, that the optimal joint policy equals $\Pi^* = I_3/3$, where I_3 is the identity matrix of size 3. Hence, the agents take multi-action (R, R), (P, P), or (S, S) with equal probability obtaining an expected reward of zero.

Optimal rank-one joint policy: Π^* (the optimal joint policy) is of rank three and thus cannot be captured by Π (an outer product of marginals). Instead, brute-force symbolic minimization, using Mathematica [3], shows that an optimal strategy for A^1 and A^2 is to let $\pi^1 = \pi^2$ with

$$\pi^1(R) = 2 - \sqrt{2} \approx 0.586, \tag{1}$$

$$\pi^1(P) = 0, \text{ and} \tag{2}$$

$$\pi^1(S) = 1 - \pi^1(R) \approx 0.414. \tag{3}$$

The expected reward from this strategy is $5 - 4\sqrt{2} \approx -.657$, far less than the optimal expected reward of 0.

A.3 Training details

Centralized agent. Fig. 6 provides an overview of the architecture of the centralized agent. The final joint policy is constructed using a single linear layer applied to a hidden state. As this architecture varies slightly when changing the number of agents and the environment (*i.e.*, AI2-THOR or our gridworld variant of AI2-THOR) we direct anyone interested in exact replication to our codebase.

AI2-THOR upgrades. As we described in Sec. 6 we have made several upgrades to AI2-THOR in order to make it possible to run our FurnMove task. These upgrades are described below.

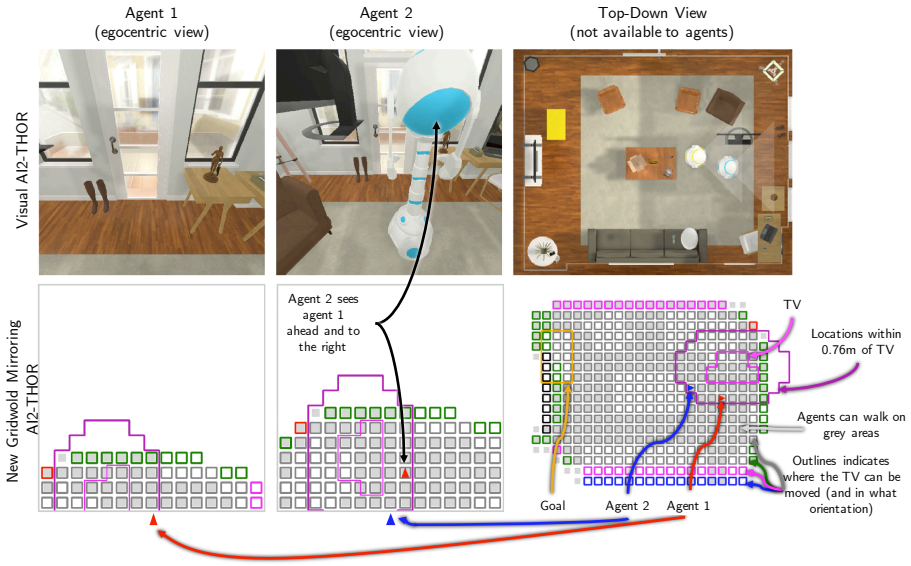

Fig. 7: **Directly comparing visual AI2-THOR with our gridworld.** The same scene with identical agent, TV, and TV-stand, positions in AI2-THOR and our gridworld mirroring AI2-THOR. Gridworld agents receive clean, task-relevant, information directly from the environment while, in AI2-THOR, agents must infer this information from complex images.

Implementing FurnMove methods in AI2-THOR's Unity codebase. The AI2-THOR simulator has been built using C# in Unity. While multi-agent support exists in AI2-THOR, our FurnMove task required implementing a collection of new methods to support randomly initializing our task and moving agents in tandem with the lifted object. Initialization is accomplished by a randomized search procedure that first finds locations in which the lifted television can be placed and then determines if the agents can be situated around the lifted object so that they are sufficiently close to the lifted object and looking at it. Implementing the joint movement actions (recall $\mathcal{A}^{\text{MWO}}$) required checking that all agents and objects can be moved along straight-line paths without encountering collisions.

Top-down Gridworld Mirroring AI2-THOR. To enable fast prototyping and comparisons between differing input modalities, we built an efficient gridworld mirroring AI2-THOR. See Fig. 7 for a side-by-side comparison of AI2-THOR and our gridworld. This gridworld was implemented primarily in Python with careful caching of data returned from AI2-THOR.

Reward structure. Rewards are provided to each agent individually at every step. These rewards include: (a) $+1$ whenever the lifted object is moved closer, in Euclidean distance, to the goal object than it had been previously in the episode, (b) a constant -0.01 step penalty to encourage short trajectories, and

(c) a penalty of -0.02 whenever the agents action fails. The minimum total reward achievable for a single agent is -7.5 corresponding to making only failed actions, while the maximum total reward equals $0.99 \cdot d$ where d is the total number of steps it would take to move the lifted furniture directly to the goal avoiding all obstructions. Our models are trained to maximize the expected discounted cumulative gain with discounting factor $\gamma = 0.99$.

Optimization and learning hyperparameters. For all tasks, we train our agents using reinforcement learning, particularly the popular A3C algorithm [6]. For FurnLift, we follow [4] and additionally use a warm start via imitation learning (DAgger [10]). When we deploy the coordination loss (CORDIAL), we modify the A3C algorithm by replacing the entropy loss with the coordination loss CORDIAL defined in Eq. (1). In our experiments we anneal the β parameter from a starting value of $\beta = 1$ to a final value of $\beta = 0.01$ over the first 5000 episodes of training. We use an ADAM optimizer with a learning rate of 10^{-4}, momentum parameters of 0.9 and 0.999, with optimizer statistics shared across processes. Gradient updates are performed in an unsynchronized fashion using a HogWild! style approach [9]. Each episode has a maximum length of 250 total steps per agent. Task-wise details follow:

- FurnMove: Visual agents for FurnMove are trained for $500,000$ episodes, across 8 TITAN V or TITAN X GPUs with 45 workers and take approximately 60 hours to train.
- Gridworld-FurnMove: Agents for Gridworld-FurnMove are trained for 1,000,000 episodes using 45 workers. Apart from parsing and caching the scene once, gridworld agents do not need to render images. Hence, we train the agents with only 1 G4 GPU, particularly the `g4dn.16xlarge` virtual machine on AWS. Agents (*i.e.*, two) for Gridworld-FurnMove take approximately 1 day to train.
- Gridworld-FurnMove-3Agents: Same implementation as above, except that agents (*i.e.*, three) for Gridworld-FurnMove-3Agents take approximately 3 days to train. This is due to an increase in the number of forward and backward passes and a CPU bottleneck. Due to the action space blowing up to $|\mathcal{A}| \times |\mathcal{A}| \times |\mathcal{A}| = 2197$ (*vs.* 169 for two agents), positive rewards become increasingly sparse. This leads to grave inefficiency in training, with no learning for $\sim$500k episodes. To overcome this, we double the positive rewards for the RL formulation for all methods within the three agent setup.
- FurnLift: We adhere to the exact training procedure laid out by Jain *et al.* [4]. Visual agents for FurnLift are trained for 100,000 episodes with the first 10,000 being warm started with a DAgger-styled imitation learning. Reinforcement learning (A3C) takes over after the warm-start period.

Integration with other MARL methods. As mentioned in Sec. 2, our contributions are orthogonal to the RL method deployed. Here we give some pointers

for integration with a deep Q-Learning and a policy gradient method.

QMIX. While we focus on policy-gradients and QMIX [8] uses Q-learning, we can formulate a SYNC for Q-Learning (and QMIX). Analogous to an actor with multiple policies, consider a value head where each agent's Q-function Q_i is replaced by a collection of Q-functions Q_i^a for $a \in A$. Action sampling is done stage-wise, i.e. agents jointly pick a strategy as $\arg\max_a Q_{SYNC}(\text{communications}, a)$, and then individually choose action $\arg\max_{u^i} Q_i^a(\tau^i, u^i)$. These Q_i^a in turn can incorporated into the QMIX mixing network.

COMA/MADDPG. Both these policy gradient algorithms utilize a centralized critic. Since our contributions focus on the actor head, we can directly replace their per-agent policy with our SYNC policies and thus benefit directly from the counterfactual baseline in COMA [2] or the centralized critic in MADDPG [5].

A.4 Quantitative evaluation details

Confidence intervals for metrics reported. In the main paper, we mentioned that we mark the best performing decentralized method in **bold** and highlight it in green if it has non-overlapping 95% confidence intervals. In this supplement, particularly in Tab. 5, Tab. 6, Tab. 7, and Tab. 8 we include the 95% confidence intervals for the metrics reported in Tab. 1, Tab. 2, Tab. 3, and Tab. 4.

Hypotheses on 3-agent *central* method performance. In Fig. 1 and Sec. 6.3 of the main paper, we mention that the *central* method performs worse than *SYNC* for the Gridworld-FurnMove-3Agent task. We hypothesize that this is because the *central* method for the -3Agent setup is significantly slower as its actor head has dramatically more parameters requiring more time to train. In numbers – the *central*'s actor head alone has $D \times |\mathcal{A}|^3$ parameters, where D is the dimensionality of the final representation fed into the actor (please see Fig. 6 for *central*'s architecture). Note, $D = 512$ for our architecture means the *central*'s actor head has $512 \cdot 13^3 = 1{,}124{,}864$ parameters. Contrast this to *SYNC*'s $D \times |\mathcal{A}| \times K$ parameters for a K mixture component. Even for the highest K in the mixture component study (Tab. 3), *i.e.*, $K = 13$, this value is $86,528$ parameters. Such a large number of parameters makes learning with the *central* agent slow even after 1M episodes (this is already $10\times$ more training episodes than used in [4]).

Why MD-SPL instead of SPL? SPL was introduced in [1] for evaluating single-agent navigational agents, and is defined as follows:

$$\text{SPL} = \frac{1}{N_{\text{ep}}} \sum_{i=1}^{N_{\text{ep}}} S_i \frac{l_i}{\max(x_i, l_i)}, \tag{4}$$

where i denotes an index over episodes, N_{ep} equals the number of test episodes, and S_i is a binary indicator for success of episode i. Also x_i is the length of the agent's path and l_i is the shortest-path distance from agent's start location to the goal. Directly adopting SPL isn't pragmatic for two reasons:

(a) Coordinating actions *at every timestep* is critical to this multi-agent task. Therefore, the number of actions taken by agents instead of distance (say in meters) should be incorporated in the metric.

(b) Shortest-path distance has been calculated for two agent systems for FurnLift [4] by finding the shortest path for each agent in a state graph. This can be done effectively for fairly independent agents. While each position of the agent corresponds to 4 states (if 4 rotations are possible), each position of the furniture object corresponds to

$$\begin{aligned} \#\text{ States} = \ & (\#\text{pos. for } A^1 \text{ near obj}) \times (\#\text{pos. for } A^2 \text{ near obj}) \\ & \times (\#\text{rot. for obj}) \times (\#\text{rot. for } A^1) \times (\#\text{rot. for } A^2), \end{aligned} \tag{5}$$

This leads to 404,480 states for an agent-object-agent assembly. We found the shortest path algorithm to be intractable in a state graph of this magnitude. Hence we resort to the closest approximation of Manhattan distance from the object's start position to the goal's position. This is the shortest path, if there were no obstacles for navigation.

Minimal edits to resolve the above two problems lead us to using actions instead of distance, and leveraging Manhattan distance instead of shortest-path distance. This leads us to defining, as described in Section Sec. 6.2 of the main paper, the Manhattan distance based SPL (MDSPL) as the quantity

$$\text{MDSPL} = \frac{1}{N_{\text{ep}}} \sum_{i=1}^{N_{\text{ep}}} S_i \frac{m_i / d_{\text{grid}}}{\max(p_i, m_i / d_{\text{grid}})}. \tag{6}$$

Defining additional metrics used for FurnLift. Jain *et al.* [4] use two metrics which they refer to as *failed pickups* (picked up, but not 'pickupable') and *missed pickups* ('pickupable' but not picked up). 'Pickupable' means when the object and agent configurations were valid for a PickUp action.

Plots for additional metrics. See Fig. 8, 9, and 10 for plots of additional metric recorded during training for the FurnMove, Gridworld-FurnMove, and FurnLift tasks. Fig. 10 in particular shows how the *failed pickups* and *missed pickups* metrics described above are substantially improved when using our SYNC models.

Additional 3-agent experiments. In the main paper we present results when training *SYNC*, *marginal*, and *central* models to complete the 3-agent Gridworld-

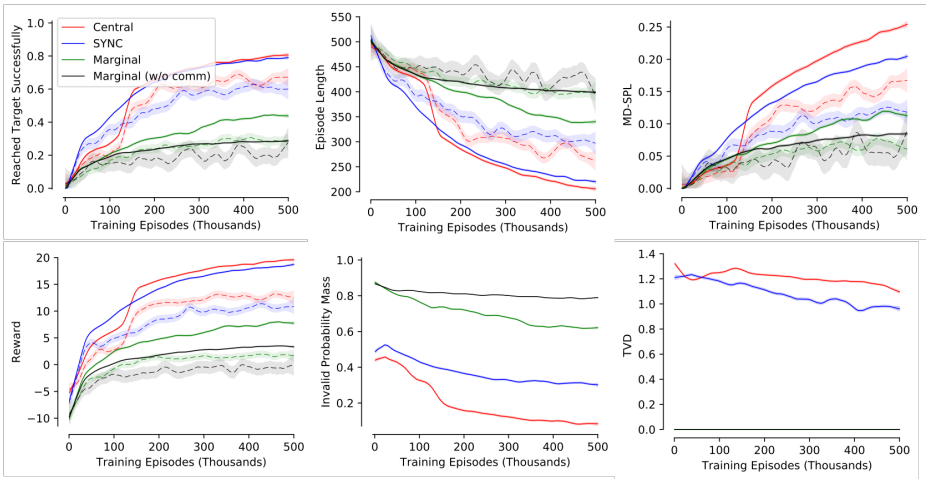

Fig. 8: **Metrics recorded during training for the** FurnMove **task for various models.** These plots add to the graph shown in Fig. 4a. Here solid lines indicate performance on the training set and dashed lines the performance on the validation set. For the *Invalid prob* and *TVD* metrics, only training set values are shown. For the *TVD* metric the black line (corresponding to the *Marginal (w/o comm)* model completely covers the green line corresponding to the *Marginal* model.

FurnMove task. We have also trained the same methods to complete the (visual) 3-agent FurnMove task. Rendering and simulating 3-agent interactions in AI2-THOR is computationally taxing. For this reason we trained our *SYNC* and *central* models for 300k episodes instead of the 500k episodes we used when training 2-agent models. As it showed no training progress, we also stopped the *marginal* model's training after 100k episodes. Training until 300k episodes took approximately four days using eight 12GB GPUs ($\sim$ 768 GPU hours per model).

After training, the *SYNC*, *marginal*, and *central* obtained a test-set success rate of $23.2 \pm 2.6\%$, $0.0 \pm 0.0\%$, and $12.4 \pm 2.0\%$ respectively. These results mirror those of the 3-agent Gridworld-FurnMove task from the main paper. Particularly, both the *SYNC* and *central* models train to reasonable success rates but the *central* model actually performs worse than the *SYNC* model. A discussion of our hypothesis for why this is the case can be found earlier in this section. In terms of our other illustrative metrics, our *SYNC*, *marginal*, and *central* respectively obtain MDSPL values of 0.029, 0.0, and 0.012, and *Invalid prob* values of 0.336, 0.854, and 0.132.

Effect of field of view (FoV). We investigate the effect of varying FoV of our agents. Unless specified otherwise, all experiments reported in this work for agents with a FoV of 90°. We additionally deploy *SYNC* with FoV of 60° and 120° to find that performance of visual agents improves with an increase

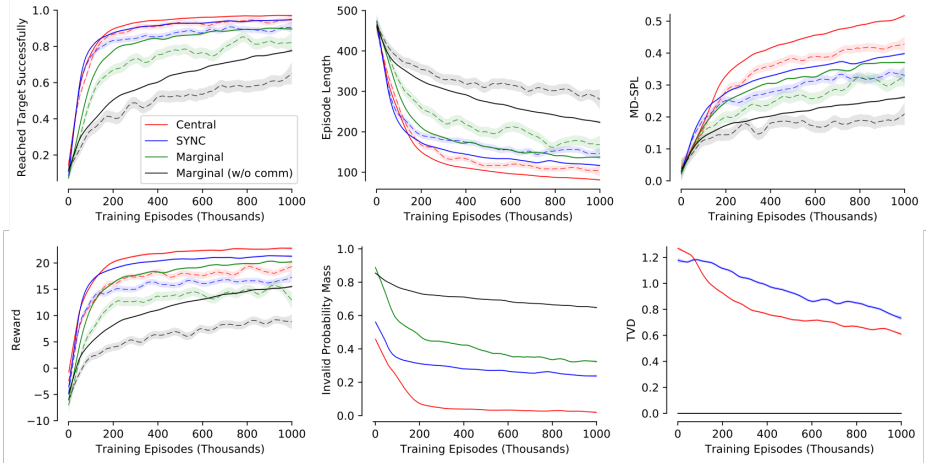

Fig. 9: **Metrics recorded during training for the Gridworld-FurnMove task for various models.** These plots add to the graph shown in Fig. 4b. Here solid lines indicate performance on the training set and dashed lines the performance on the validation set. For the *Invalid prob* and *TVD* metrics, only training set values are shown. For the *TVD* metric the black line (corresponding to the *Marginal (w/o comm)* model completely covers the green line corresponding to the *Marginal* model.

in FoV. Particularly, we observe a success rate of 0.538, 0.587, and 0.661 for *SYNC* with 60°, 90°, and 120°, respectively. As we do for other visual multi-agent experiments, we trained agents to 500k episodes (60 hrs on 8 TITAN X GPUs) for this study.

A.5 Qualitative evaluation details and a statistical analysis of learned communication

Discussion of our qualitative video (https://youtu.be/I_Evs5Bol6k). This video includes four clips, each corresponding to the rollout on a test scene of one of our models trained to complete the FurnMove task.

Clip A. *Marginal* agents attempt to move the TV to the goal but get stuck in a narrow corridor as they struggle to successfully coordinate their actions. The episode is considered a failure as the agents do not reach the goal in the allotted 250 timesteps. A top-down summary of this trajectory is included in Fig. 11.

Clip B. Unlike the *marginal* agents from Clip A., in this clip two *SYNC* agents successfully coordinate actions and move the TV to the goal location in 186 steps. A top-down summary of this trajectory is included in Fig. 12.

Clip C. Here we show *SYNC* agents completing the Gridworld-FurnMove in a test scene (the same scene and initial starting positions as in Clip A and Clip B). The agents complete the task in 148 timesteps even after an initial search in the incorrect direction.

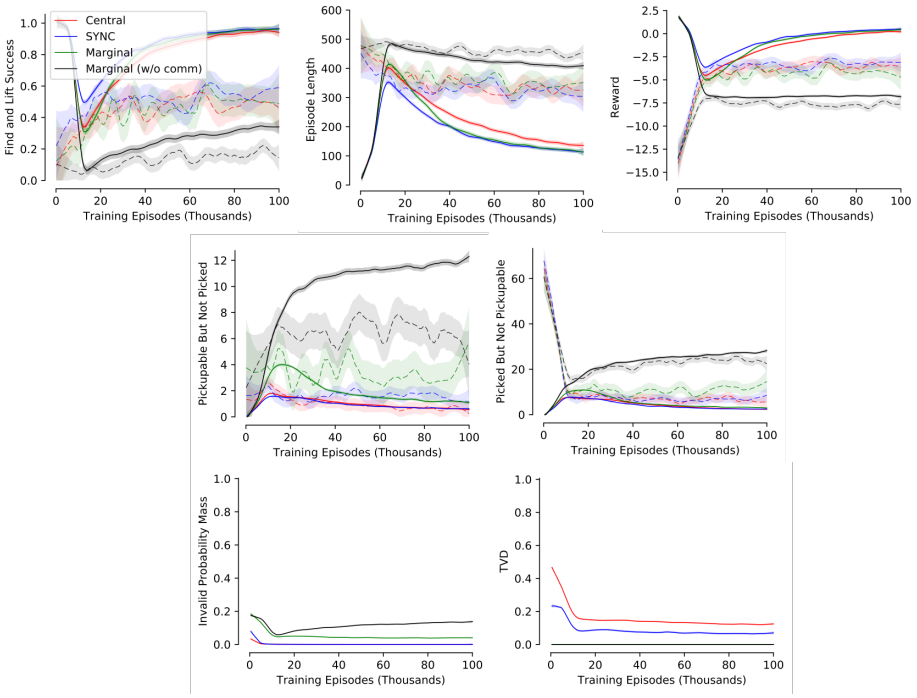

Fig. 10: **Metrics recorded during training for the** FurnLift **task for various models.** These plots add to the graph shown in Fig. 4c. Notice that we have included plots corresponding to the *failed pickups* (picked up, but not 'pickupable') and *missed pickups* ('pickupable' but not picked up) metrics described in Sec. A.4. Solid lines indicate performance on the training set and dashed lines the performance on the validation set. For the *Invalid prob* and *TVD* metrics, only training set values are shown. For the *TVD* metric the black line (corresponding to the *Marginal (w/o comm)* model completely covers the green line corresponding to the *Marginal* model.

Clip D (contains audio). This clip is an attempt to *experience* what agents 'hear.' The video for this clip is the same as Clip B showing the *SYNC* method. The audio is a rendering of the communication between agents in the reply stage. Particularly, we discretize the $[0, 1]$ value associated with the first reply weight of each agent into 128 evenly spaced bins corresponding to the 128 notes on a MIDI keyboard (0 corresponding to a frequency of $\sim$8.18 Hz and 127 to $\sim$12500 Hz). Next, we post-process the audio so that the communication from the agents is played on different channels (stereo) and has the Tech Bass tonal quality. As a result, the reader can experience what agent 1 hears (*i.e.*, agent 2's reply weight) via the left earphone/speaker and what agent 2 hears (*i.e.*, agent 1's reply weight) via the right speaker. In addition to the study in Sec. 6.4 and Sec. A.5, we notice a higher pitch/frequency for the agent which is passing.

Table 5: 95% confidence intervals included in addition to Tab. 1, evaluating methods on FurnMove, Gridworld-FurnMove, and Gridworld-FurnMove-3Agents. For legend details, see Tab. 1.

Methods	MD-SPL ↑	Success ↑	Ep len ↓	Final dist ↓	Invalid prob. ↓	TVD ↕
FurnMove (ours)						
Marginal w/o comm [4]	0.032 (±0.007)	0.164 (±0.023)	224.1 (±2.031)	2.143 (±0.104)	0.815 (±0.005)	0 (±0)
Marginal [4]	0.064 (±0.008)	0.328 (±0.029)	194.6 (±2.693)	1.828 (±0.105)	0.647 (±0.010)	0 (±0)
SYNC	**0.114** (±0.009)	**0.587** (±0.031)	**153.5** (±2.870)	**1.153** (±0.089)	**0.31** (±0.004)	0.474 (±0.005)
Central†	0.161 (±0.012)	0.648 (±0.030)	139.8 (±2.958)	0.903 (±0.076)	0.075 (±0.006)	0.543 (±0.006)
Gridworld-FurnMove (ours)						
Marginal w/o comm [4]	0.111 (±0.012)	0.484 (±0.031)	172.6 (±2.825)	1.525 (±0.121)	0.73 (±0.008)	0 (±0)
Marginal [4]	0.218 (±0.015)	0.694 (±0.029)	120.1 (±2.974)	0.960 (±0.100)	0.399 (±0.011)	0 (±0)
SYNC	**0.228** (±0.014)	**0.762** (±0.026)	**110.4** (±2.832)	**0.711** (±0.076)	**0.275** (±0.005)	0.429 (±0.005)
Central†	0.323 (±0.016)	0.818 (±0.024)	87.7 (±2.729)	0.611 (±0.067)	0.039 (±0.004)	0.347 (±0.006)
Gridworld-FurnMove-3Agents (ours)						
Marginal [4]	0 (±0)	0 (±0)	250.0 (±0)	3.564 (±0.111)	0.823 (±0)	0 (±0)
SYNC	**0.152** (±0.012)	**0.578** (±0.031)	**149.1** (±6.020)	**1.05** (±0.091)	**0.181** (±0.006)	0.514 (±0.009)
Central†	0.066 (±0.008)	0.352 (±0.03)	195.4 (±5.200)	1.522 (±0.099)	0.138 (±0.005)	0.521 (±0.006)

We also notice lower pitches for MoveWithObject and MoveObject actions.

Joint policy summaries. These provide a way to visualize the effective joint distribution that each method captures. For each episode in the test set, we log each multi-action attempted by a method. We average over steps in the episode to obtain a matrix (which sums to one). Afterwards, we average these matrices (one for each episode) to create a *joint policy summary* of the method for the entire test set. This two-staged averaging prevents the snapshot from being skewed towards actions enacted in longer (failed or challenging) episodes. In the main paper, we included snapshots for FurnMove in Fig. 5. In Fig. 13 we include additional visualizations for all methods including (*Marginal w/o comm* model) for FurnMove and Gridworld-FurnMove.

Communication analysis. As shown in Fig. 5d and discussed in Sec. 6.4, there is very strong qualitative evidence suggesting that our agents use their talk and reply communication channels to explicitly relay their intentions and coordinate their actions. We now produce a statistical, quantitative, evaluation of

Table 6: 95% confidence intervals included in addition to Tab. 2, evaluating methods on FurnLift. *Marginal* and *SYNC* perform equally well, and mostly lie within confidence intervals of each other. *Invalid prob.* and *failed pickups* metrics for *SYNC* have non-overlapping confidence bounds (lighted in green). For more details on the legend, see Tab. 1.

Methods	MD-SPL ↑	Success ↑	Ep len ↓	Final dist ↓	Invalid prob. ↓	TVD ↕	Failed pickups ↓	Missed pickups ↓
FurnLift [4] ('constrained' setting with no implicit communication)								
Marginal w/o comm [4]	0.029 (±0.007)	0.15 (±0.022)	229.5 (±3.482)	2.455 (±0.105)	0.11 (±0.004)	0 (±0)	25.219 (±1.001)	6.501 (±0.784)
Marginal [4]	**0.145** (±0.016)	**0.449** (±0.031)	**174.1** (±5.934)	2.259 (±0.094)	0.042 (±0.003)	0 (±0)	8.933 (±0.867)	1.426 (±0.284)
SYNC	0.139 (±0.016)	0.423 (±0.031)	176.9 (±5.939)	**2.228** (±0.083)	**0** (±0)	0.027 (±0.002)	**4.873** (±0.453)	**1.048** (±0.192)
Central†	0.145 (±0.016)	0.453 (±0.031)	172.3 (±5.954)	2.331 (±0.088)	0 (±0)	0.059 (±0.002)	5.145 (±0.5)	0.639 (±0.164)

Table 7: 95% confidence intervals included in addition to Tab. 3 by varying number of components in SYNC-policies for FurnMove.

K in SYNC	MD-SPL ↑	Success ↑	Ep len ↓	Final dist ↓	Invalid prob. ↓	TVD ↕
FurnMove						
1 component	0.064 (±0.004)	0.328 (±0.019)	194.6 (±2.833)	1.828 (±0.105)	0.647 (±0.002)	0 (±0)
2 components	0.084 (±0.008)	0.502 (±0.031)	175.5 (±5.321)	1.227 (±0.091)	**0.308** (±0.004)	0.206 (±0.004)
4 components	0.114 (±0.009)	0.569 (±0.031)	154.1 (±5.783)	**1.078** (±0.083)	0.339 (±0.004)	0.421 (±0.005)
13 components	**0.114** (±0.009)	**0.587** (±0.031)	**153.5** (±5.739)	1.153 (±0.089)	0.31 (±0.004)	0.474 (±0.005)

this phenomenon by fitting multiple logistic regression models where we attempt to predict, from the agents communications, certain aspects of their environment as well as their future actions. In particular, we run 1000 episodes on our test set using our mixture model in the visual testbed. This produces a dataset of 159,380 observations where each observation records, for a single step by both agents at time t:

(a) The two weights $p^1_{\text{talk},t}, p^2_{\text{talk},t}$ where $p^i_{\text{talk},t}$ is the weight agent A^i assigns to the first symbol in the "talk" vocabulary.
(b) The two weights $p^1_{\text{reply},t}, p^2_{\text{reply},t}$ where $p^i_{\text{reply},t}$ is the weight agent A^i assigns to the first symbol in the "reply" vocabulary.
(c) The two values $\texttt{tv}^i_t \in \{0, 1\}$ where $\texttt{tv}^i_t$ equals 1 if and only if agent A^i sees the TV at timestep t (before taking its action).
(d) The two values $\texttt{WillPass}^i_t \in \{0, 1\}$ where $\texttt{WillPass}^i_t$ equals 1 if and only if agent i ends up choosing to take the Pass action at time t (*i.e.*, after finishing communication).
(e) The two values $\texttt{WillMWO}^i_t \in \{0, 1\}$ where $\texttt{WillMWO}^i_t$ equals 1 if and only if agent i ends up choosing to take some MoveWithObject action at time t.

Table 8: 95% confidence intervals included in addition to Tab. 4, ablating coordination loss on *marginal* [4], *SYNC*, and *central* methods. †denotes that a centralized agent serve only as an upper bound to decentralized methods.

Method	CORDIAL	**MD-SPL** ↑	**Success** ↑	**Ep len** ↓	**Final dist** ↓	**Invalid prob.** ↓	**TVD** ↕
				FurnMove			
Marginal	✗	0.064 (±0.008)	0.328 (±0.029)	194.6 (±5.385)	1.828 (±0.105)	0.647 (±0.01)	0 (±0.0)
Marginal	✓	0.015 (±0.004)	0.099 (±0.019)	236.9 (±2.833)	2.134 (±0.105)	0.492 (±0.002)	0 (±0.0)
SYNC	✗	0.091 (±0.008)	0.488 (±0.031)	170.3 (±5.665)	1.458 (±0.104)	0.47 (±0.008)	0.36 (±0.008)
SYNC	✓	0.114 (±0.009)	0.587 (±0.031)	153.5 (±5.739)	1.153 (±0.089)	0.31 (±0.004)	0.474 (±0.005)
Central†	✗	0.14 (±0.011)	0.609 (±0.03)	146.9 (±5.895)	1.018 (±0.084)	0.155 (±0.006)	0.6245 (±0.005)
Central†	✓	0.161 (±0.012)	0.648 (±0.03)	139.8 (±5.915)	0.903 (±0.076)	0.075 (±0.006)	0.543 (±0.006)

Fig. 11: Clip A trajectory summary. The marginal agents quickly get stuck in a narrow area between a sofa and the wall and fail to make progress.

In the following we will drop the subscript t and consider the above quantities as random samples drawn from the distribution of possible steps taken by our agents in randomly initialized trajectories. As A^1 and A^2 share almost all of their parameters they are, essentially, interchangeable. Because of this our following analysis will be solely taking the perspective of agent A^1, similar results hold for A^2. We consider fitting the three models:

$$\begin{aligned}\sigma^{-1}P(\texttt{tv}_t^1 = 1) = \beta_{\text{tv}} &+ \beta^1_{\text{talk, tv}} \cdot p^1_{\text{talk}} \\ &+ \beta^1_{\text{reply, tv}} \cdot p^1_{\text{talk}}, \\ &+ \beta^1_{\text{talk*reply, tv}} \cdot p^1_{\text{talk}} \cdot p^1_{\text{reply}},\end{aligned} \tag{7}$$

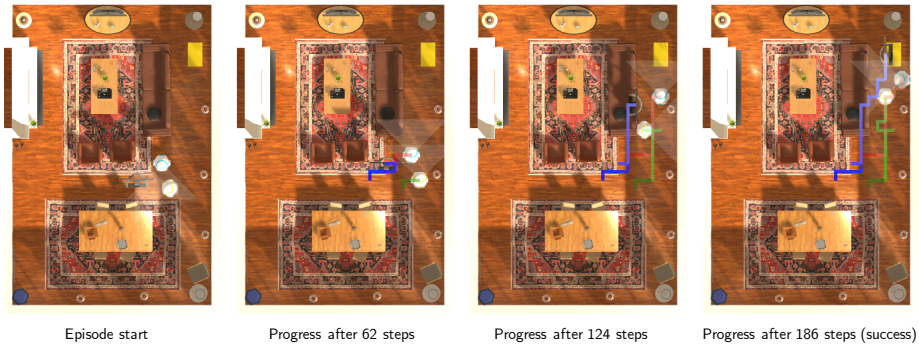

Fig. 12: Clip B trajectory summary. The SYNC agents successfully navigate the TV to the goal location without getting stuck in the narrow corridor.

Table 9: Estimates, and corresponding robust bootstrap standard errors, for the parameters of communication analysis (Sec. A.5).

	β_{tv}	$\beta^1_{\text{talk, tv}}$	$\beta^1_{\text{reply, tv}}$	$\beta^1_{\text{talk*reply, tv}}$	-
Est.	-2.62	6.93	3.35	-8.44	-
SE	0.33	0.52	0.38	0.62	-

	β_{pass}	$\beta^1_{\text{talk, pass}}$	$\beta^2_{\text{talk, pass}}$	$\beta^1_{\text{reply, pass}}$	$\beta^2_{\text{reply, pass}}$
Est.	-7.55	2.69	-2.2	-1.72	9.98
SE	0.09	0.09	0.08	0.07	0.11

	β_{MWO}	$\beta^1_{\text{talk, MWO}}$	$\beta^2_{\text{talk, MWO}}$	$\beta^1_{\text{reply, MWO}}$	$\beta^2_{\text{reply, MWO}}$
Est.	2.71	0.39	0.28	-3.34	-3.37
SE	0.05	0.06	0.06	0.06	0.06

$$\sigma^{-1}P(\texttt{WillPass}^1 = 1) = \beta_{\text{pass}} + \sum_{i=1}^{2} \beta^i_{\text{talk, pass}} \cdot p^i_{\text{talk}} + \sum_{i=1}^{2} \beta^i_{\text{reply, pass}} \cdot p^i_{\text{reply}}, \text{and} \tag{8}$$

$$\sigma^{-1}P(\texttt{WillMWO}^1 = 1) = \beta_{\text{MWO}} + \sum_{i=1}^{2} \beta^i_{\text{talk, MWO}} \cdot p^i_{\text{talk}} + \sum_{i=1}^{2} \beta^i_{\text{reply, MWO}} \cdot p^i_{\text{reply}}, \tag{9}$$

where σ is the usual logistic function. Here Eq. (7) attempts to determine the relationship between what A^1 communicates and whether or not A^1 is currently seeing the TV, Eq. (8) probes whether or not any communication symbol is

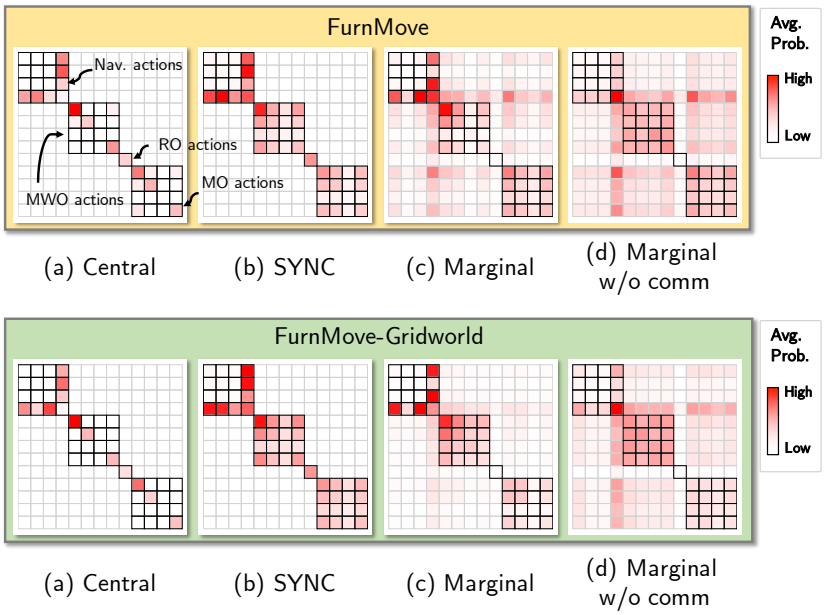

Fig. 13: **Additional results for Fig. 5.** Joint policy summaries for all methods for both FurnMove and Gridworld-FurnMove.

associated with A^1 choosing to take a Pass action, and finally Eq. (9) considers whether or not A^1 will choose to take a MoveWithObject action. We fit each of the above models using the `glm` function in the R programming language [7]. Moreover, we compute confidence intervals for our coefficient values using a robust bootstrap procedure. Fitted parameter values can be found in Tab. 9.

From Tab. 9 we draw several conclusions. First, in our dataset, there is a somewhat complex association between agent A^1 seeing the TV and the communication symbols it sends. In particular, for a fixed reply weight $p^1_{\text{reply}} < 0.821$, a larger value of p^1_{talk} is associated with higher odds of the TV being visible to A^1 but if $p^1_{\text{reply}} > 0.821$ then larger values of p^1_{talk} are associated with smaller odds of the TV being visible. When considering whether or not A^1 will pass, the table shows that this decision is strongly associated with the value of p^2_{reply} where, given fixed values for the other talk and reply weights, p^2_{reply} being larger by a unit of 0.1 is associated with 2.7× larger odds of A^1 taking the pass action. This suggests the interpretation of a large value of p^2_{reply} as A^2 communicating that it wishes A^1 to pass so that A^1 may perform a single-agent navigation action to reposition itself. Finally, when considering the fitted values corresponding to Eq. (9) we see that while the talk symbols communicated by the agents are weakly related with whether or not A^1 takes a MoveWithObject action, the reply symbols are associated with coefficients with an order of magnitude larger values. In particular, assuming all other communication values are fixed,

a smaller value of either p^1_{reply} or p^2_{reply} is associated with substantially larger odds of A^1 choosing a MoveWithObject action. This suggests interpreting an especially small value of p^i_{reply} as agent A^i indicating its readiness to move the object.

References

1. Anderson, P., Chang, A., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., et al.: On evaluation of embodied navigation agents. arXiv preprint arXiv:1807.06757 (2018)
2. Foerster, J.N., Farquhar, G., Afouras, T., NArdelli, N., Whiteson, S.: Counterfactual Multi-Agent Policy Gradients. In: AAAI (2018)
3. Inc., W.R.: Mathematica, Version 12.1, https://www.wolfram.com/mathematica, champaign, IL, 2020
4. Jain*, U., Weihs*, L., Kolve, E., Rastegari, M., Lazebnik, S., Farhadi, A., Schwing, A.G., Kembhavi, A.: Two body problem: Collaborative visual task completion. In: CVPR (2019), * equal contribution
5. Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I.: Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In: NeurIPS (2017)
6. Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning. In: ICML (2016)
7. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2019), https://www.R-project.org/
8. Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., Whiteson, S.: Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In: ICML (2018)
9. Recht, B., Re, C., Wright, S., Niu, F.: Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In: NeurIPS (2011)
10. Ross, S., Gordon, G., Bagnell, D.: A reduction of imitation learning and structured prediction to no-regret online learning. In: AISTATS (2011)