Zero Shot Coordination for Sparse Reward Tasks with Diverse Reward Shapings

Keenan Powell [1] **Peihong Yu** [1] **Pratap Tokekar** [1]

Abstract

Many Multi-Agent Reinforcement Learning (MARL) agents fail to adapt properly to cooperating with agents trained with the same objectives and goals but different seeds, algorithms, or other training differences. This is the problem of Zero-Shot Coordination (ZSC), which focuses on training agents to cooperate well with unknown agents. ZSC has been studied for a variety of tabular cases and simple games such as Hanabi, achieving excellent results. However, existing solutions to ZSC only consider identical rewards for your trained agents and all future partners. This is not realistic for the trained agents, as they do not consider the problem of cooperating with agents that have identical sparse objectives but shape the rewards for those objectives in different manner. To address this issue, we show how to train an ensemble of methods using randomized reward shapings chosen using 4 selection algorithms. Experiments done on the Overcooked environment demonstrate consistent improvements of 62.2%-119.2% in sparse reward over baseline ZSC algorithms when playing with agents that have identical sparse rewards but different reward shapings.

1. INTRODUCTION AND RELATED WORK

Multi-Agent Reinforcement Learning (MARL) has emerged as a powerful tool for training agents capable of coordination across fields such as vehicle routing, self-driving vehicles, cooperative games, and multi-drone pursuit (Liu et al., 2023; Foerster et al., 2019; Sessa et al., 2020; Li et al., 2025; Hu et al., 2021). However, traditional MARL algorithms and techniques often fail when attempting to cooperate with unknown partners even under indentical objectives and goals. Because multiple optimal strategies can exist simultaneously under Self-Play (Tesauro, 1994), agents can easily form conventions and pre-coordinated strategies that don't work when attempting to coordinate with agents that have formed different conventions and strategies. Zero-Shot Coordination (ZSC) addresses this issue by training agents to form a policy that can adapt to a wide variety of other partners (in this paper, other trained agents) without requiring additional training with the new partner. While many existing ZSC approaches have achieved excellent results, including Other-Play (Hu et al., 2021), which takes advantage of symmetries in the environment, TrajeDi (Lupu et al., 2021), which uses ensemble learning with a diversity component in the loss, and MAZE (Xue et al., 2024), which co-evolves heterogenous populations, these approaches generally either assume or rely upon identical reward functions between different agents. This is unrealistic: while cooperating agents will likely share sparse objectives, their reward shapings (weights for environmental features to densify a sparse reward) (Ibrahim et al., 2024), will often vary depending on where and how they were trained. For example, self-driving cars from multiple companies may share a sparse objective of reaching a destination but weight aspects

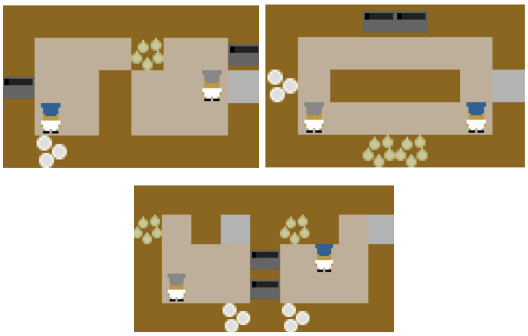

Figure 1. Three Overcooked Environments. Pictured in top left is Random0_Medium, top right is Random3, and bottom is Unident_S. Included are onions, pots, plates, stoves, and a delivery station. How would the blue and gray agents cooperate to the best extent possible without pre-coordination or knowing each other's strategies? How would we identify good performances across all 3 environments, including where strong cooperation is forced, like in Random0_Medium or Unident_S?

[1]Department of Computer Sceicne, University of Maryland, College Park, USA. Correspondence to: Keenan Powell <kpowell1@terpmail.umd.edu>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

like speed vs. safety differently. Hidden-Utility Self-Play (HSP) (Yu et al., 2023) one of the more similar algorithms to ours, extends the standard self-play framework for zero-shot cooperation by explicitly modeling human biases as hidden reward functions. HSP generates a diverse policy pool by randomly sampling feature weights to produce agents with varied biases. While HSP can produce different rewards, it is meant for specific human biases based on an existing environmental reward, and does not expand well to the more general idea of ZSC between agents, or other forms of reward shaping. It also relies on purely random feature weighting, without any enforcement of diversity or performance.

There has been additionally been some research done on on-the-fly adaptation to completely unknown agents (Zand et al., 2022; Fuchs et al., 2021; Liu et al., 2023; Sessa et al., 2020). These have found strong results, being able to learn the behaviors of partner and opposing agents over many iterations of play, but on-the-fly adaptation is a less demanding problem than ZSC, where you are expected to learn the best policy over multiple evaluation episodes instead of being able to cooperate well immediately.

We address this gap in considered policies by training ZSC agents with diverse reward shapings. We propose 4 novel methods of selecting these diverse sets of reward shapings (LLM-Based, Surrogate Network, Stratified Grid, and Random) and create ensemble models from populations trained on different shapings. Unlike existing ZSC methods that create diverse policies within a single reward function or randomly alter it slightly, this allows our trained agents to more accurately adapt to other agents trained with different reward shapings by creating diversity across both policy space and reward functions. We evaluate our method on the Overcooked (Carroll et al., 2020) environment, where results show 62.2%-119.2% improvement in sparse reward for the best selection methods compared to the baseline algorithms when cooperating with agents trained with unknown reward shapings.

Our contributions are as follows:

- We introduce 4 shaping selection methods and an ensemble framework for combining ZSC models.
- We demonstrate that Stratified Grid sampling and MLP-based selection achieve the best generalization to unknown partners with unknown reward shapings.

2. Problem Definition

We formulate the ZSC problem in terms of a decentralized partially observable Markov decision process (Dec-POMDP). A Dec-POMDP $\mathbb{M}$ is defined as the tuple $\mathbb{M} = (\mathbb{N}, \mathbb{S}, \mathbb{A}, \mathbb{T}, R, \mathbb{O}, \gamma, \mu)$, where $\mathbb{N}$ is the set of agents, representing $N = |\mathbb{N}|$ cooperative agents, $\mathbb{S}$ is the joint state space shared by each agent, $\mathbb{A} = \prod_{k \in \mathbb{N}} a_k$ is the joint action space, where a_k is the specific action space for agent k, $\mathbb{T} : \mathbb{S} \times \mathbb{A} \rightarrow \Delta\mathbb{S}$ is the transition function which determines the distribution of the next state given the current joint state and action, $R : \mathbb{S} \times \mathbb{A} \rightarrow \mathbb{R}$ is the reward function, $\mathbb{O}$ is the observation function, γ is the discount factor, and μ is the initial state distribution.

The objective of a policy π in a Dec-POMDP is to maximize the expected return over finite time horizon H:

$$\pi^* = \text{argmax}_{\pi \in \Pi} J(\pi)$$

where

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{H} \gamma^t R(s_t, a_t) \middle| \pi \right]$$

where $\Pi = \{\pi = (\pi_1 ... \pi_N) | \pi^i = o_t^i \rightarrow a_i^t\}$ is the set of decentralized policies, with each individual policies selecting action a^i based on their local observation o^i.

The goal for ZSC is to learn a joint policy that generalizes well during cooperation with unknown agents and unknown policies, trained with different methods. Unlike traditional training and self-play algorithms, agents will not know each others policies during training or testing, meaning that conventions and pre-coordination attempts will fail. This gives rise to a specific performance metric for ZSC, the average cross-play (XP) loss, which for the two player case is given as:

$$J_{XP}(\pi_1, \pi_2) = \frac{1}{2}(J(\pi_1^1, \pi_2^2) + J(\pi_2^1, \pi_1^2))$$

where π^1 and π^2 represent the individual components of the policies (i.e. which player they are playing as).

3. Selecting Diverse Reward Shapings

We aim to improve the generalization of ZSC and expand it to a wider context by considering the problem of both diverse reward shapings and diverse policies. We introduce methods for selecting reward shapings that are then used to generate a population of models during training, and finally evaluated together as an ensemble model with unknown partner agents that have identical sparse objectives but unknown reward shapings.

3.1. Selection and Usage

For our purposes, we consider a reward shaping as a set of weights, each referring to a specific environmental feature. We use the following notation for reward shapings and our selection methods: sparse(s_t, a_t) is the sparse reward in state s_t with action a_t, f_k is a specific environmental feature used for reward shaping, w_k is the weight applied to feature f_k, $P_i = (w_1; ...; w_k)$ is the reward shaping, a vector of

weights applied to the features, $F = |P_i|$ is the number of features used, and P is the number of reward shapings generated by the selection method. This gives us a notation for the densification of a reward as

$$R(s_t, a_t) = \text{sparse}(s_t, a_t) + \sum_{k=1}^{F} w_k f_k$$

The notion of densifying the reward with a weighted combination of features is the most common way of reward shaping, and is inbuilt into many MARL environments. We create and evaluate 4 methods of selecting reward shapings:

LLM-Based: Inspired by (Ma et al., 2024), we utilize LLMs to generate reward shapings. Specifically, we prompt Claude Sonnet 4.5 with a list of randomly generated $P_1 ... P_n$ (n is arbitrary, we used 41 examples) as well as their individual training and testing results in terms of both sparse and shaped reward. We also supplied the code for both the environment and the agent, and asked it to generate a new set P that would both perform well and result in an ending population of diverse policies and agents. Exact prompt and example output is given in Table 9.

Surrogate Network: We train a Multi-Layer Perceptron (MLP) using supervised learning on the same examples given to the LLM. Specifically, for model $g(\theta)$ we input reward shaping P_i and expect output total sparse reward sparse_i. This model then evaluates 1000 randomly generated shapings and then we select the best set of predicted results. Exact details on architecture are given in A.

Stratified Grid: We use Latin Hypercube Sampling (McKay et al., 1979) to select a group of reward shapings. This works as a form of stratified sampling on each feature present. We divide the range of possible weights for each feature f_k into P strata of equal probability $1/P$ and sample once from each stratum. Calling that sample X_k, and performing the same process for each feature gives us an array of weights $w_{k,j}$, where k is the associated feature. We then form random P_i by selecting one unused weight from each column. This ensures that all possible weights are evenly represented and gives us a group of P_i that reasonably represents the entire grid of possible reward shapings.

Random: We uniformly randomly select each weight w_k from the possible range for feature f_k to form P_i. This gives us a reasonably diverse set of reward shapings, but without any guarantees or knowledge of how they might perform or whether they accurately represent the space.

3.2. Policy Population Generation

To actually implement these selection methods, we build off of the Trajectory Diversity (TrajeDi) method (Lupu et al., 2021), which trains a population of models with diverse policies by implementing a Jensen-Shannon divergence term within the reward function. Specifically, TrajeDi trains a population of n agents as well as an $(n+1)$-th agent to act as a Best Response (BR) to each other agent in the population, which is used as the final output of the training. The loss for TrajeDi is formulated as

$$\mathbb{L}(BR, \pi_1, ..., \pi_n) = -\Big[J(BR) + \alpha JSD_\gamma(\pi_1, ..., \pi_n) + \sum_{i=1}^{n}(J_{XP}(BR, \pi_i) + J(\pi_i))\Big]$$

where $JSD(\pi_1, ..., \pi_n)$ is the Jensen-Shannon divergence (Lin, 1991).

To generate our final agents, we use each selection method to generate P reward shapings. We then train a population based on each of those unique shapings with the TrajeDi algorithm, and then combine these populations by using each BR agent as a single component of a final ensemble model, with the action chosen by the ensemble model being the mode action:

$$\pi_{\text{ensemble}}(o_i) = \text{argmax}_{a_i \in A} \sum_{i=1}^{N} 1[BR_i(o_i) = a_i]$$

where $1[]$ is the indicator function.

To compare this formulation with the baseline TrajeDi algorithm, we also compare an ensembled formulation of the baseline, equivalent to doing the same process as above but selecting the same base reward shaping each time.

We compare both the original baseline, the ensembled baseline, and our selection methods by evaluating them in cooperation with agents that have been trained using randomly selected unknown reward shapings. These other agents are trained simply using MAPPO (Yu et al., 2022), a baseline MARL algorithm not made for ZSC. We also compare all these algorithms with Hidden-Utility Self-Play (HSP) (Yu et al., 2023), explained further in the Related Work.

4. Experimental Setup

We conduct experiments on the Overcooked (Carroll et al., 2020) environment. This is based on the cooking game of the same name, a cooperative game where player are given tasks to cook various dishes within a short time limit, having to perform various tasks such as cooking, chopping, and combining different ingredients. The goal in Overcooked is to deliver as many orders as possible, with delivering orders ahead of time giving you bonus score, and failing to deliver orders on time applying penalties. Overcooked has become one of the most popular environments for evaluating the coordination and cooperation of MARL agents due to the diversity of tasks and ease of use (Wang et al., 2024; Yan et al., 2023). We utilize three specific environments

within Overcooked - Random0_Medium, Random3, and Unident_S, all pictured in Figure 1. We select these three due to their range of difficulty - Random3 has all agents in the same layout, with no forced strong cooperation, while Random0_Medium and Unident_S either strongly encourage or force cooperation, with distinct separated areas and asymmetric resources.

We use the code from ZSC-Eval (Wang et al., 2024), which is a pre-built library for training and evaluating ZSC algorithms, including existing code for both the Overcooked environment and TrajeDi. We use $P = 10$ reward shapings and train our agents with 100 million timesteps and then evaluate them over 10 seeds, each with 40 rollouts that each take 40000 timesteps. During training, we also perform a short evaluation of the models every 20 million timesteps. Note that these short evaluations during training are with agents that have the same reward shapings.

We consider each reward shaping element has having a possible weight from 0 to 10. The reward shaping elements we used are as follows: PLACEMENT_IN_POT_REW, which rewards the agent for placing an ingredient in a pot, DISH_PICKUP_REWARD, which rewards the agent for picking up a dish, SOUP_PICKUP_REWARD, which rewards the agent for picking up soup, DISH_DISP_DISTANCE_REW, which rewards the agent for being close to the dish dispenser, POT_DISTANCE_REW, which rewards the agent for being close to a pot, and SOUP_DISTANCE_REW, which rewards the agent for being close to soup.

We aim to answer three questions: Does introducing diverse reward shapings improve the performance over the baseline algorithm? Which selection methods produce the most diverse reward shapings? Is diversity of the reward shapings correlated with performance?

5. Results

Algorithm	Sparse Reward	vs Base. (ens.)
LLM-Based	104.1 ± 1.58	+54.3%
Surrogate Network	123.2 ± 1.24	+82.7%
Stratified Grid	**131.4 ± 1.82**	+94.9%
Random	85.6 ± 1.69	+27.0%
HSP (Yu et al., 2023)	26.8 ± 3.17	-60.2%
Baseline (Lupu et al., 2021)	20.6 ± 6.21	-69.5%
Baseline (ensembled)	67.4 ± 0.64	—

Table 1. Sparse Reward comparison during evaluation on Random3 environment. Values show mean ± 90% CI.

Algorithm	Shaped Reward	vs Base. (ens.)
LLM-Based	90.1 ± 1.10	+61.1%
Surrogate Network	103.0 ± 0.97	+84.2%
Stratified Grid	**107.4 ± 1.08**	+92.2%
Random	72.5 ± 1.10	+29.6%
HSP (Yu et al., 2023)	24.4 ± 2.65	-56.6%
Baseline (Lupu et al., 2021)	20.9 ± 4.78	-62.6%
Baseline (ensembled)	55.9 ± 0.39	—

Table 2. Shaped Reward comparison during evaluation on Random3 environment. Values show mean ± 90% CI.

Algorithm	Sparse Reward	vs Base. (ens.)
LLM-Based	47.1 ± 0.78	+30.1%
Surrogate Network	**59.3 ± 0.86**	+63.8%
Stratified Grid	58.7 ± 0.36	+62.2%
Random	49.3 ± 0.74	+36.2%
HSP (Yu et al., 2023)	14.8 ± 1.53	-59.2%
Baseline (Lupu et al., 2021)	15.0 ± 3.55	-58.5%
Baseline (ensembled)	36.2 ± 0.60	—

Table 3. Sparse Reward comparison on Random0_Medium environment. Values show mean ± 90% CI.

Algorithm	Shaped Reward	vs Base. (ens.)
LLM-Based	**146.1 ± 0.66**	+100.0%
Surrogate Network	141.5 ± 0.48	+93.6%
Stratified Grid	136.0 ± 0.33	+86.0%
Random	94.2 ± 0.46	+29.0%
HSP (Yu et al., 2023)	38.8 ± 6.71	-46.9%
Baseline (Lupu et al., 2021)	32.6 ± 8.40	-55.4%
Baseline (ensembled)	73.1 ± 0.30	—

Table 4. Shaped Reward comparison on Random0_Medium environment. Values show mean ± 90% CI.

Algorithm	Sparse Reward	vs Base. (ens.)
LLM-Based	72.5 ± 0.99	+102.5%
Surrogate Network	**78.5 ± 1.04**	+119.2%
Stratified Grid	69.7 ± 0.88	+94.7%
Random	54.2 ± 1.09	+51.4%
HSP (Yu et al., 2023)	24.9 ± 6.39	-30.4%
Baseline (Lupu et al., 2021)	22.5 ± 8.59	-37.2%
Baseline (ensembled)	35.8 ± 0.61	—

Table 5. Sparse Reward comparison on Unident_s environment. Values show mean ± 90% CI.

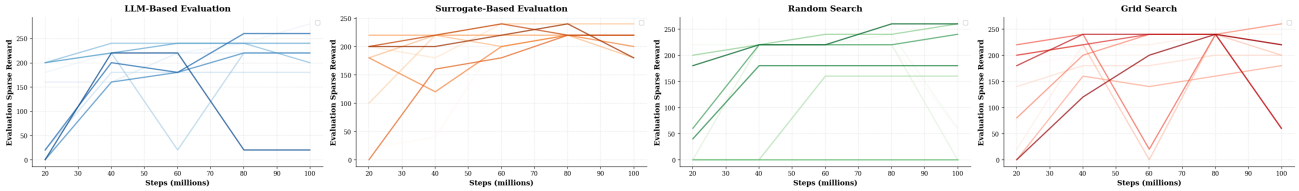

Figure 2. Evaluation runs during training. Each line shows the performance of a BR agent in one of the populations made by the selection method. Displayed is the sparse reward, evaluated once every 20 million timesteps during the 100 million training timesteps.

Algorithm	Shaped Reward	vs Base. (ens.)
LLM-Based	56.9 ± 0.65	+44.1%
Surrogate Network	$\mathbf{67.8 \pm 0.56}$	+71.6%
Stratified Grid	66.6 ± 0.82	+68.6%
Random	51.7 ± 0.61	+30.9%
HSP (Yu et al., 2023)	$\underline{20.4 \pm 4.45}$	-48.4%
Baseline (Lupu et al., 2021)	24.2 ± 9.7	-39.7%
Baseline (ensembled)	39.5 ± 0.83	—

Table 6. Shaped Reward comparison on Unident_s environment. Values show mean $\pm$ 90% CI.

Selection Method	Avg. Stdev	% Range Covered
LLM-Based	3.0558	83.3
Surrogate Network	2.2926	63.3
Stratified Grid	2.9452	96.7
Random	3.2352	91.7

Table 7. Diversity metrics for each selection method. Averaged Standard Deviation is the average of the standard deviation of each weight across the group of reward shapings. % Range covered is the average of range covered (within the possible 0-10) by the values associated with each weight.

5.1. Performance

Seen in Table 1 through Table6, our algorithms improve upon the baseline in all cases and environments. Specifically, Surrogate Network and Stratified Grid exhibit the most significant improvements, seeing anywhere from a $\sim$60% improvement to a $\sim$120% improvement across sparse and shaped rewards, and almost always performing the best across all methods. We can see that a significant improvement is also obtained from the ensemble nature of the methods, with the ensembled version of TrajeDi generally exhibiting an 40-70% improvement over the non-ensembled version. This is likely partially due to the fact that these evaluations were performed against models using unknown reward shapings and the more robust, adaptable, and resilient properties of ensemble models. However, we can clearly see that not all improvement is exhibited to that, with the ensembled Baseline always being improved by our methods with at least 27% improvement. The poor performance of random selection vs. all other methods is likely due to the fact that no guarantees or attempts were made on random to either improve the performance or diversity like with all other methods, but even so, it improves on the baselines by a significant amount.

5.2. Diversity

Shown in Table 7 (and with more specificity in Table 8) we can see the averaged standard deviation across the produced weights for each selection method. That is, taking the standard deviation for the values given to each weight, and then averaging those values across all the weights. These values are then averaged across all 3 environments to display in the table above. Notable is that even though the Surrogate Network selection maintains a significant standard deviation for values 0-10, it is significantly lower than all other selection methods, by around 0.65. All three other selection methods hover around a much closer range, from about $2.95 - 3.25$. One thing of note is that performance increases as standard deviation decreases between Stratified Grid, LLM-Based, and Random selection, but Surrogate Network selection performance decreases even as the standard deviation decreases. It's likely that the most optimal variance lays between the approaches of Surrogate Network (most closely optimal and based on existing data) and Stratified Grid (ignoring existing data and giving the most varied data reasonably possible).

Standard deviation is not the only method of diversity - we can also see in Table 7 that Stratified Grid covers the largest range of each weight on average, slightly higher than random selection, even though Random selection has a higher standard deviation. This is due to the fact that Random Selection can easily select multiple reward shapings that are very similar to each other, creating results that clump and do not accurately reflect across the entire space. Surrogate Network performs excellently despite it's low coverage due to picking rewards that have achieved the highest sparse reward in the past, as with 1000 rewards to choose from it is likely to consistently pick rewards most similar to those that have performed excellently, and even if those are not especially diverse, their individual performances and reasonable coverage can likely make up for it. Even though both methods were given the exact same data, Surrogate Network

likely picks a much smaller range of values compared to LLM-Based due to the LLM being prompted specifically to choose a set of shapings that would result in diverse policies.

Seen in Figure 2 is the evaluation steps run during training time on the Random3 environment, showing the performance of each BR agent for each reward shaping selected by the selection method. Surrogate Network converges to a small set of excellently performing policies, showing both its lack of diversity and its weight on individual performance due to how it was trained and used. LLM-Based is similar, but with a slightly wider range, and 1 policy that performs badly in the ending two evaluation steps, likely due to the differences mentioned above. Stratified Grid shows the most evenly spaced out performance, with most reward shapings performing reasonably well, but not being concentrated in a specific area or shape like the model-based selections. Random has performances clumped into three areas, with 3 reward shapings demonstrating very poor performance over time, likely due to both it's high diversity but lack of constraints on representative diversity.

6. Conclusion and Future Work

Introducing multiple reward shapings vastly improves performance: Even with the most naive method of selection, we still find significant improvements over using the same reward shaping for each agent.

Our methods can generate diverse reward shapings: Our methods generated significantly more diverse methods than sticking closely to one reward shaping set, with even the least diverse method covering almost 2/3 of the space.

Diversity is useful, but not the only factor of importance: While the most successful method was Stratified Grid sampling, which is the most representative and reasonably diverse, the second most successful method was Surrogate Network selection, the least diverse and representative.

Future work will focus on expanding the breadth of testing, expanding to more environments and more baseline algorithms. In addition, modifying the training and prompting of the Surrogate Network and LLM may yield increased performance for those methods. Specifically, introducing a diversity metric to encourage the Surrogate Network to evaluate a more representative set of samples higher may improve performance. This may allow us to take the best aspects from both Surrogate Network and Stratified Grid sampling.

References

Carroll, M., Shah, R., Ho, M. K., Griffiths, T. L., Seshia, S. A., Abbeel, P., and Dragan, A. On the utility of learning about humans for human-ai coordination, 2020. URL https://arxiv.org/abs/1910.05789.

Foerster, J. N., Song, F., Hughes, E., Burch, N., Dunning, I., Whiteson, S., Botvinick, M., and Bowling, M. Bayesian action decoder for deep multi-agent reinforcement learning, 2019. URL https://arxiv.org/abs/1811.01458.

Fuchs, A., Walton, M., Chadwick, T., and Lange, D. Theory of mind for deep reinforcement learning in hanabi, 2021. URL https://arxiv.org/abs/2101.09328.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Teh, Y. W. and Titterington, M. (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/glorot10a.html.

Hu, H., Lerer, A., Peysakhovich, A., and Foerster, J. "other-play" for zero-shot coordination, 2021. URL https://arxiv.org/abs/2003.02979.

Ibrahim, S., Mostafa, M., Jnadi, A., Salloum, H., and Osinenko, P. Comprehensive overview of reward engineering and shaping in advancing reinforcement learning applications, 2024. URL https://arxiv.org/abs/2408.10215.

Li, Y., Chen, J., Xue, F., Qiu, J., Li, W., Zhang, Q., Wen, Y., and Pan, W. At-drone: Benchmarking adaptive teaming in multi-drone pursuit, 2025. URL https://arxiv.org/abs/2502.09762.

Lin, J. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991. doi: 10.1109/18.61115.

Liu, X., Peters, L., and Alonso-Mora, J. Learning to play trajectory games against opponents with unknown objectives, 2023. URL https://arxiv.org/abs/2211.13779.

Lupu, A., Cui, B., Hu, H., and Foerster, J. Trajectory diversity for zero-shot coordination. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7204–7213. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/lupu21a.html.

Ma, Y. J., Liang, W., Wang, G., Huang, D.-A., Bastani, O., Jayaraman, D., Zhu, Y., Fan, L., and Anandkumar, A. Eureka: Human-level reward design via coding large language models, 2024. URL https://arxiv.org/abs/2310.12931.

McKay, M. D., Beckman, R. J., and Conover, W. J. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979. ISSN 00401706. URL http://www.jstor.org/stable/1268522.

Sessa, P. G., Bogunovic, I., Kamgarpour, M., and Krause, A. Learning to play sequential games versus unknown opponents. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 8971–8981. Curran Associates, Inc., 2020.

Tesauro, G. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Comput.*, 6 (2):215–219, March 1994. ISSN 0899-7667. doi: 10.1162/neco.1994.6.2.215. URL https://doi.org/10.1162/neco.1994.6.2.215.

Wang, X., Zhang, S., Zhang, W., Dong, W., Chen, J., Wen, Y., and Zhang, W. Zsc-eval: An evaluation toolkit and benchmark for multi-agent zero-shot coordination, 2024. URL https://arxiv.org/abs/2310.05208.

Xue, K., Wang, Y., Guan, C., Yuan, L., Fu, H., Fu, Q., Qian, C., and Yu, Y. Heterogeneous multi-agent zero-shot coordination by coevolution, 2024. URL https://arxiv.org/abs/2208.04957.

Yan, X., Guo, J., Lou, X., Wang, J., Zhang, H., and Du, Y. An efficient end-to-end training approach for zero-shot human-ai coordination. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of ppo in cooperative, multi-agent games, 2022. URL https://arxiv.org/abs/2103.01955.

Yu, C., Gao, J., Liu, W., Xu, B., Tang, H., Yang, J., Wang, Y., and Wu, Y. Learning zero-shot cooperation with humans, assuming humans are biased, 2023. URL https://arxiv.org/abs/2302.01605.

Zand, J., Parker-Holder, J., and Roberts, S. J. On-the-fly strategy adaptation for ad-hoc agent coordination, 2022. URL https://arxiv.org/abs/2203.08015.

A. Surrogate Network Architecture

Due to the small amount of examples (just 41), we utilized a small model (6 inputs with two hidden layers each with 32 and 16 units), as well as 30% dropout and Xavier initialization (Glorot & Bengio, 2010). We train for 500 epochs with 0.001 learning rate and a batch size of 8 to prevent overfitting.

B. MARL Hyperparameters

We use the default Hyperparameters given used for ZSC-Eval (Wang et al., 2024). This includes a γ of 0.99, learning rate of 0.0005, and clip_param of 0.2.

C. Qualitative Analysis for LLM-Based and Surrogate Network

Parameter	**LLM-Based**		**Surrogate Network**		**Stratified Grid**		**Random**	
	Mean	**Stdev**	**Mean**	**Stdev**	**Mean**	**Stdev**	**Mean**	**Stdev**
PLACEMENT_IN_POT_REW	7.7	2.00	9.6	1.29	3.5	3.21	4.9	3.31
DISH_PICKUP_REWARD	6.0	3.80	9.5	1.65	5.3	3.13	4.8	3.79
SOUP_PICKUP_REWARD	4.4	2.95	9.7	2.00	5.0	2.87	4.5	2.95
DISH_DISP_DISTANCE_REW	3.9	2.96	5.0	3.86	3.8	2.78	5.0	3.30
POT_DISTANCE_REW	4.1	3.03	1.3	1.83	5.9	3.14	4.0	3.68
SOUP_DISTANCE_REW	3.8	3.58	6.7	3.13	5.2	4.29	7.6	2.37
Average	**5.0**	**3.06**	**7.0**	**2.29**	**4.8**	**3.24**	**5.1**	**3.24**

Table 8. More specific mean and standard deviation for selection methods.

We can see a more detailed breakdown of variables in Table 8 compared to the results in the main paper. Random has a couple outliers, but in general is very close to what you would expect, around 5.0 mean and around 3.0 standard deviation. Stratified Grid has a few more outliers, but still hovers around the 4-6 range with around 3.0 standard deviation. LLM-Based and Surrogate Network show more interesting results, with a lot of values (especially for Surrogate Network) abnormally high or with significantly lower standard deviations. This is likely due to the best performing reward shapings being ones that are strongly shaped, as that allows the model to densify the reward to a very detailed extent.

D. LLM Prompts

Displayed in Table 9 is the specific prompt used for the LLM to generate new reward shapings. You can see the specific emphasis on creating diverse shapings and how that played in to the LLMs rationale. You can also see how the LLM attempts to create specific strategies for each reward shaping cofiguration.

Table 9. The following table contains an example of the prompt used for the LLM and its response.

System: Attached are two files.
1: Part of the code for the Overcooked reinforcement learning environment, which has multiple possible shaped rewards.
2: Training results from training populations of agents using specific reward shaping weights.

I want you to propose to me, given the attached 2 files, a set of 10 reward shaping weights (using the same 6 shaped infos as the results file). I want you to prioritize two things.
1: The diversity of your proposed reward shaping weights (not making them too similar to each other). You would prefer to propose a set of 10 weights that will create a set of 10 diverse policies.
2: The estimated performances of your proposed weights.

Propose to me a format like this: [params..., params..., ...]

[Attached results file:]

[

”folder”: ”traj-S1-s10-4-2-random”,
”pop_num”: 4,
”seed”: 2,
”run_type”: ”random”,
”best_reward”: 468.9725875854492,
”reward_shaping_params”:
”PLACEMENT_IN_POT_REW”: 10,
”DISH_PICKUP_REWARD”: 8,
”SOUP_PICKUP_REWARD”: 2,
”DISH_DISP_DISTANCE_REW”: 3,
”POT_DISTANCE_REW”: 5,
”SOUP_DISTANCE_REW”: 0
,
...
]

[Attached Overcooked Environment Code]

LLM:
json[
{
”PLACEMENT_IN_POT_REW”: 9,
”DISH_PICKUP_REWARD”: 10,
”SOUP_PICKUP_REWARD”: 7,
”DISH_DISP_DISTANCE_REW”: 9,
”POT_DISTANCE_REW”: 0,
”SOUP_DISTANCE_REW”: 2
},
...
]
Rationale:
High performers selected: I prioritized configurations from the top-performing runs (537, 505, 481, 475, 468, 464, 457, 454, 450 reward range)
Diverse behavioral strategies:
Configs 1, 3, 6: Emphasize early-game actions (placement + dish pickup) with varying distance rewards
Config 2, 7, 8: Focus heavily on pot placement with minimal soup delivery shaping
Config 4: Balanced approach with strong soup pickup emphasis
Config 5, 10: Low dish pickup, encouraging different coordination patterns
Config 9: Medium values across the board for stable coordination
Key diversity dimensions:
DISH_PICKUP_REWARD: ranges from 0 to 10 (creates agents with different item prioritization)
Distance rewards: varied combinations to encourage different movement patterns PLACEMENT vs SOUP_PICKUP trade-offs: different temporal focuses in the task pipeline

This set should produce diverse policies while maintaining strong performance potential.