PIMAEX: Multi-Agent Exploration Through Peer Incentivization

Michael Kölle, Johannes Tochtermann, Julian Schönberger, Gerhard Stenzel, Philipp Altmann and Claudia Linnhoff-Popien
Institute of Informatics, LMU Munich, Munich, Germany
fi

Keywords: Multi-Agent Reinforcement Learning, Intrinsic Curiosity, Social Influence.

Abstract: While exploration in single-agent reinforcement learning has been studied extensively in recent years, considerably less work has focused on its counterpart in multi-agent reinforcement learning. To address this issue, this work proposes a peer-incentivized reward function inspired by previous research on intrinsic curiosity and influence-based rewards. The *PIMAEX* reward, short for Peer-Incentivized Multi-Agent Exploration, aims to improve exploration in the multi-agent setting by encouraging agents to exert influence over each other to increase the likelihood of encountering novel states. We evaluate the *PIMAEX* reward in conjunction with *PIMAEX-Communication*, a multi-agent training algorithm that employs a communication channel for agents to influence one another. The evaluation is conducted in the *Consume/Explore* environment, a partially observable environment with deceptive rewards, specifically designed to challenge the exploration vs. exploitation dilemma and the credit-assignment problem. The results empirically demonstrate that agents using the *PIMAEX* reward with *PIMAEX-Communication* outperform those that do not.

1 INTRODUCTION

One of the main challenges in Reinforcement Learning (RL) is the exploration vs. exploitation dilemma; that is, an RL agent must find a suitable trade-off between exploratory and exploitative behavior to avoid getting stuck in local optima. This is especially important for hard exploration problems, which often exhibit sparse or deceptive rewards; that is, rewards may occur rarely or be misleading. This issue is also related to another important problem in RL, the credit-assignment problem: if a long sequence of actions without any direct reward must be taken to eventually obtain a reward, RL agents might fail to assign credit to those non-rewarding actions that are temporally distant from the eventual reward.

In such problems, naive approaches based purely on random exploration, such as ϵ-greedy policies, often fail to learn successful policies. Consequently, many approaches have been proposed to tackle these challenges. While much work in single-agent RL has focused on intrinsic curiosity rewards and novelty of encountered states to aid exploration, there is considerably less literature aimed specifically at multi-agent exploration. This is likely because the state space of multi-agent RL (MARL) systems grows exponentially with the number of agents, making exploration in this setting a much harder problem than in single-agent RL.

Inspired by previous work in intrinsic curiosity and influence-based rewards, this work proposes a peer-incentivization scheme, in which an agent rewards its peers for influencing it to discover novel states. Accordingly, the main contribution of this work is the formulation of a multi-agent social influence peer reward function, the *PIMAEX* reward, aimed at improving exploration in multi-agent settings with sparse or deceptive rewards. Additionally, a multi-agent Reinforcement Learning algorithm employing this reward, *PIMAEX-Communication*, is introduced. The *PIMAEX* reward is a specific instance of a generalized multi-agent social influence peer reward function, also introduced in this work, comprising three terms α, β, and γ. The α term is essentially the influence reward introduced by (Jaques et al., 2018), while the γ term is comparable to that in (Wang et al., 2019) and was part of the proposed future work in (Jaques et al., 2018). Therefore, the contribution lies in the β term, which, to the best of the author's knowledge, has not yet been proposed, as well as in the generalized formulation combining all three terms in a weighted sum.

To evaluate *PIMAEX-Communication*, this work uses the *Consume/Explore* environment, a partially

Copyright © 2025 by Paper published under CC license (CC BY-NC-ND 4.0)

observable multi-agent environment with deceptive rewards, designed specifically to address the exploration vs. exploitation dilemma and the credit-assignment problem.

Section 2 reviews research on intrinsic curiosity, influence-based rewards, and multi-agent exploration. Section 3 introduces the *PIMAEX* reward and algorithm. Section 5 outlines the experimental setup. Section 6 compares results with baselines. Section 7 summarizes contributions and future work.

2 RELATED WORK

The *PIMAEX* reward function is inspired by two main areas in reinforcement learning: intrinsic curiosity rewards, which encourage exploration, and influence-based rewards, where agents receive rewards based on the impact of their actions on peers. This section explores these areas, with intrinsic curiosity rewards in Section 2.1 and influence-based rewards in Section 2.3. Influence-based rewards naturally apply to multi-agent reinforcement learning, where they assist with coordination and cooperation. An overview of the related cooperative MARL approaches is given in Section 2.2, focusing on methods most relevant to the mechanisms proposed in this work.

2.1 Intrinsic Motivation: Curiosity

Intrinsic curiosity rewards are widely used to drive exploration, particularly in challenging environments. These rewards supplement or sometimes replace the environment's reward by incentivizing agents to seek novelty. Two common methods are count-based exploration and prediction-error exploration. Count-based methods compute novelty by counting state visits, e.g., by giving a reward proportional to $\frac{1}{\sqrt{N(s)}}$. However, this approach is feasible only in small state spaces and relies on approximations, such as density models or hash functions, in larger spaces (Bellemare et al., 2016; Ostrovski et al., 2017; Tang et al., 2016).

Prediction-error methods, introduced by Schmidhuber (Schmidhuber, 1991), reward agents based on the error of a learned model predicting future states. High prediction errors signify novel states, making this method effective for exploration. Variants of this approach use forward dynamics models to predict next states (Oudeyer et al., 2007; Stadie et al., 2015) or inverse dynamics models to avoid uncontrollable environmental factors (Pathak et al., 2017). To overcome issues like the "noisy TV problem" (Burda et al., 2018), where agents get attracted to random, high-error stimuli, Burda et al. propose *Random Network Distillation* (*RND*) (Burda et al., 2018). In *RND*, a randomly initialized neural network serves as the target for a second network to predict, with prediction errors used as curiosity rewards. This method is computationally light, but requires observation and reward normalization to avoid inconsistencies (Burda et al., 2018).

2.2 Multi-Agent Cooperation and Coordination

Influence-based rewards are part of broader MARL approaches that promote agent cooperation. Many methods leverage centralized training and decentralized execution (CTDE), sharing Q-networks across agents and decomposing centralized Q-functions (Fu et al., 2022; Foerster et al., 2017; Rashid et al., 2018). Communication between agents is also common for improved coordination (Peng et al., 2017; Sukhbaatar et al., 2016). Another approach involves counterfactual reasoning to determine individual agent contributions in the absence of explicit individual rewards, as in (Foerster et al., 2017). Peer incentivization, where agents can reward or penalize others, is a relevant direction (Yang et al., 2020; Schmid et al., 2021).

2.3 Social Influence

In settings where agents maximize their own rewards, social influence can encourage collaboration without a central reward. Jaques et al. (Jaques et al., 2018) propose rewarding agents based on the influence they exert on other agents' policies, measured via counterfactual reasoning. They evaluate influence by conditioning one agent's policy on another's actions and comparing it with a counterfactual scenario where the influence is removed. This discrepancy quantifies influence and encourages coordination by maximizing mutual information.

In Jaques et al.'s experiments, agents either influence others through discrete message communication or use models to predict others' actions. In the latter, agents employ a Model of Other Agents (MOA) to relax the need for centralized training. They note that social influence reduces policy gradient variance, which can increase with the number of agents (Lowe et al., 2017).

Influence-Based Multi-Agent Exploration

Wang et al. (Wang et al., 2019) address limitations in single-agent curiosity by proposing two approaches: exploration via information-theoretic influence (EITI) and exploration via decision-theoretic influence (EDTI). EITI uses mutual information to quantify how an agent's actions affect others' learning trajectories, while EDTI introduces the Value of Interaction (VoI), which evaluates the long-term influence of one agent on another's expected return, including both extrinsic and intrinsic factors. They achieve this using neural networks to approximate transition dynamics for large state spaces, thereby allowing EITI and EDTI to be applied to complex environments.

3 PEER-INCENTIVIZED MULTI-AGENT EXPLORATION

In multi-agent RL, exploration is significantly more challenging than in single-agent scenarios because the joint state space grows exponentially with the number of agents. As state transitions depend on joint actions, it is improbable that one agent alone can cover much of the state space, so coordinated multi-agent exploration is often essential. Although many works address single-agent exploration or multi-agent coordination, relatively few focus specifically on multi-agent exploration (Section 2).

Building on prior work in intrinsic curiosity and influence-based rewards, we introduce the *Peer Incentivized Multi-Agent Exploration* (*PIMAEX*) reward function (Section 3.1). *PIMAEX* rewards an agent for influencing others to visit novel or rarely visited states. It is a specific instance of a generalized multi-agent social influence reward function (also introduced in Section 3.1), general enough to encompass approaches from (Jaques et al., 2018; Wang et al., 2019) and to allow various influence measures and communication channels.

To demonstrate this in practice, Section 4 presents *PIMAEX-Communication*, a multi-agent training algorithm inspired by (Jaques et al., 2018). It employs a communication channel so agents can send messages, influencing one another's behavior. Counterfactual reasoning marginalizes the influence of each agent's message on others, enabling the *PIMAEX* reward. As *PIMAEX-Communication* can work with any *actor-critic* algorithm, we focus on the modifications needed for the communication channel, counterfactual reasoning, and the *PIMAEX* reward rather than on policy and value updates. These modifications affect agents' neural network inference functions and the acting and learning components of the training loop, discussed in Section 4.

3.1 Multi-Agent Social Influence Reward Functions

We unify prior concepts (Jaques et al., 2018; Wang et al., 2019) into a generalized social influence reward. Within this framework, the *PIMAEX* reward encourages exploration of novel states by combining influence measures with intrinsic curiosity. Two influence types are included: policy influence (PI), akin to causal influence in (Jaques et al., 2018), and value influence (VI), similar to the Value of Interaction (VoI) in (Wang et al., 2019), detailed in Section 3.1.1.

3.1.1 Policy and Value Influence

Let $info_{j\to i}$ denote information from agent j available to agent i at time t (e.g., past actions, observations, messages). An informed policy π_i^{info} and value function V_i^{info} for agent i use both o_i and $info_{j\to i}$. The marginal policy $\pi_{j\to i}^{marginal}$ and value $V_{j\to i}^{marginal}$ exclude $info_{j\to i}$, reflecting how i would behave if uninfluenced by j.

Marginal policies and values are computed by replacing $info_{j\to i}$ with counterfactuals $info_{j\to i}^{cf}$ and averaging to remove its effect. Policy influence (PI) measures the divergence between π_i^{info} and $\pi_{j\to i}^{marginal}$, while value influence (VI) is:

$$VI_{j\to i} = V_i^{info} - V_{j\to i}^{marginal} \tag{1}$$

Using D_{KL}, policy influence is:

$$PI_{j\to i}^{D_{KL}} = D_{KL}\left[\pi_i^{info} | \pi_{j\to i}^{marginal}\right], \tag{2}$$

and using *PMI*,

$$PI_{j\to i}^{PMI} = \log \frac{p(a_i, |, o_i, info_{j\to i})}{p(a_i, |, o_i)}. \tag{3}$$

3.1.2 Reward Functions

This unified social influence reward integrates direct influence and the long-term value of that influence. Following (Jaques et al., 2018; Wang et al., 2019), agent j's reward is:

$$r_j = \sum_{k\neq j} \left[\alpha \cdot PI_{j\to k}^{\alpha} + \beta \cdot PI_{j\to k}^{\beta} \cdot r_k^{\mathrm{w}} + \gamma \cdot VI_{j\to k}^{\mathrm{w}}\right], \tag{4}$$

where α, β, γ weight each term; $PI_{j\to k}^{\alpha}$ and $PI_{j\to k}^{\beta}$ are influence measures like D_{KL} or PMI; $VI_{j\to k}^{\mathrm{w}}$ is weighted value influence; and r_k^{w} is a weighted reward stream for agent k.

PIMAEX uses both extrinsic and intrinsic rewards within the weighted influence terms:

$$r_k^{\mathrm{w}} = \beta^{\mathrm{env}} \cdot r_k^{\mathrm{env}} + \beta^{\mathrm{int}} \cdot r_k^{\mathrm{int}}\, VI_{j\to k}^{\mathrm{w}} \tag{5}$$

$$= \gamma^{\mathrm{env}} \cdot VI_{j\to k}^{\mathrm{env}} + \gamma^{\mathrm{int}} \cdot VI_{j\to k}^{\mathrm{int}}. \tag{6}$$

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

4 PIMAEX-COMMUNICATION

PIMAEX-Communication is a MARL algorithm inspired by (Jaques et al., 2018), where agents use discrete communication policies and values. At each timestep, agents emit discrete messages, forming a communication channel for mutual influence. Counterfactual reasoning is then employed to isolate the effect of each agent's message on others, enabling the *PIMAEX* reward. *PIMAEX-Communication* can be combined with any *actor-critic* approach, so details of policy and value updates remain part of the underlying method. Here, we focus on modifications needed for communication, counterfactual reasoning, and the *PIMAEX* reward.

4.1 Network Architecture

Two main modifications are required to implement *PIMAEX-Communication*. First, each agent must have a communication policy and value head, plus an additional value head for intrinsic rewards. Second, agents must input the joint communication observation (the concatenated message vector) to their networks. Following (Jaques et al., 2018), this message vector is concatenated with environment features in the last shared layer for all policy and value heads, and an embedding layer is added between this shared layer and the communication heads.

For marginal policy/value calculations, let N be the number of agents, M the counterfactual messages per agent, and B the batch size. This creates $(N-1)\times M$ counterfactual observations per forward pass. We compute the environment features once, stack M copies, and merge batch dimensions for efficient batched calculations.

4.2 Actor and Learner

At each timestep, a *PIMAEX-Communication* actor computes three value estimates (extrinsic, intrinsic, and communication) and samples actions for both communication and environment policies. For each agent, M counterfactual communication actions are also sampled (e.g., all possible discrete messages except the one taken). These counterfactual vectors can be centrally constructed or built individually by each agent. Influence is computed on the actor side; intrinsic and *PIMAEX* rewards are computed on the learner side, which uses *Random Network Distillation* (*RND*)(Burda et al., 2018) to normalize intrinsic rewards. A one-step delay between communication actions and *PIMAEX* rewards aligns with (Jaques et al., 2018). We use a weighted sum of environment, intrinsic, and *PIMAEX* rewards for the communication policy; the environment policy uses environment and intrinsic rewards only.

4.3 Technical Implementation Details

Our *PIMAEX-Communication* implementation is built on *acme*(Hoffman et al., 2020) and *JAX*(Bradbury et al., 2018). Because *acme*'s *PPO* implementation (Schulman et al., 2017) does not allow multiple value functions or reward streams, we created a version permitting separate policies, values, and weighted rewards. This flexible implementation supports *PPO*, *PPO+RND*, and *PIMAEX-Communication*, allowing distinct hyperparameters per reward, value, or policy. We also adopted observation and reward normalization for *RND* as recommended by (Burda et al., 2018).

5 EXPERIMENTAL SETUP

This section outlines the setup used to evaluate our approach. We first introduce the *Consume/Explore* environment (Section 5.1), a partially observable multi-agent task designed to test the exploration-exploitation dilemma and credit assignment. Then, we detail agent configurations, including hyperparameters and network architectures, followed by our evaluation methodology in Section 5.2, which covers performance measures collected during training and inference.

5.1 Consume/Explore Environment

The *Consume/Explore* environment challenges agents with partial observability, a deceptive reward, and a sequential social dilemma(Leibo et al., 2017), where agents can cooperate or defect. Each of the N agents owns a production line yielding C items every M steps, stored in a depot with capacity $S_{\max}$. If the depot is full, production pauses until space becomes available. Agents start with S_{init} items, and the parameters $M, C_{\text{init}}, C_{\max}, S_{\text{init}}, S_{\max}$ control resource abundance.

Agents have three actions: do nothing, consume for reward R (if items are available), or explore to eventually increase C but gain no immediate reward. Increasing C requires $c_{\max}$ successful explore actions; success depends on E (the coordination threshold) and the number of simultaneous explore actions. Larger $c_{\max}$ increases credit-assignment difficulty. Unsuccessful exploration can incur a penalty P.

Our experiments used four agents per environment, with $c_{\max}$ tuned so that increasing C by more than two levels requires teamwork and $E = 0.5$ demanding at least two agents to explore simultaneously. Each agent's five-element observation vector includes three private elements (current supply, depot capacity status, time to next yield) and two global elements (current C and the count c toward $C+1$). All values are normalized to $[0,1]$.

5.2 Methodology

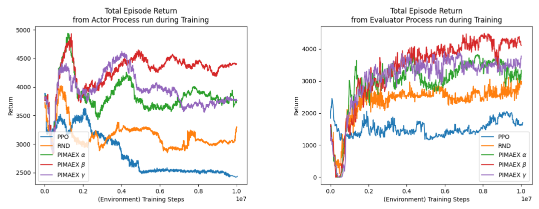
We evaluated *PIMAEX-Communication* alongside two baselines: 'vanilla' PPO and PPO+RND (using Random Network Distillation(Burda et al., 2018)). First, we ran exploratory training with 'vanilla' PPO to identify a challenging environment configuration and used its hyperparameters (Table 2 in the Appendix) as a starting point. Next, we searched RND-specific hyperparameters for PPO+RND (Table 3 in the Appendix), then applied the best settings to all *PIMAEX-Communication* agents. To isolate the effects of each *PIMAEX* term, we also trained 'single-term' agents using only one of α, β, or γ.

Each model was trained under three random seeds, with results averaged. We evaluated inference performance over 600 episodes per model (200 per seed), also averaged. Key metrics include joint return (to gauge team performance), individual returns (to see division of labor), action statistics (consume vs. explore), state space coverage (measure of exploration), and production yield (indicating cooperation). Faster progression to higher yield levels implies stronger coordination.

6 RESULTS

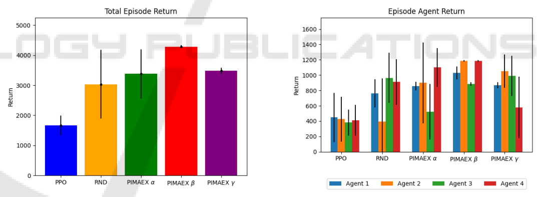
We compare the best-performing models of each agent class: 'vanilla' PPO, PPO with RND intrinsic curiosity rewards (abbreviated as RND), and 'single-term' *PIMAEX* agents (*PIMAEX* α, *PIMAEX* β, and *PIMAEX* γ). Hyperparameter settings for these models are given in Table 5 in the Appendix. As mentioned in the previous section, agent performance is assessed using various measures, focusing on exploration behavior.

Fig. 1 displays the total episode return (joint return of all agents) over training. The left figure shows return for actor processes (agents act stochastically), and the right shows the evaluator process (agents act greedily). 'Vanilla' PPO agents fail to learn a successful policy, performing worse as training progresses. In contrast, curious agents (PPO+RND and *PIMAEX* agents) initially prioritize maximizing intrinsic return, resulting in low extrinsic return early on, likely due to higher prediction error in the RND model at the start of training. However, this does not hinder long-term performance: PPO+RND outperforms 'vanilla' PPO, and is itself outperformed by 'single-term' *PIMAEX* agents, with *PIMAEX* β being the best-performing method.

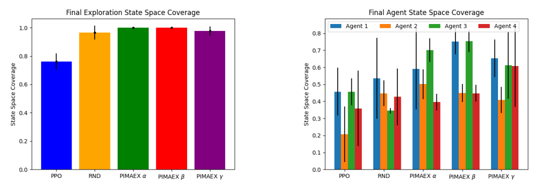


(a) Actor processes return (b) Evaluator process return
Figure 1: Overall return per episode for best training run of each agent category.

These trends are confirmed in Fig. 2, which shows mean and standard deviation of total joint episode return and per-agent individual return at inference time. Again, 'single-term' *PIMAEX* agents, with *PIMAEX* β as the best, outperform PPO+RND, which outperforms 'vanilla' PPO. Notably, *PIMAEX* β exhibits significantly less standard deviation than other methods, a consistent pattern across all performance metrics.



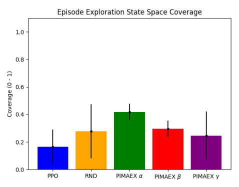
(a) Overall episode return (b) Per-agent episode return
Figure 2: Per-episode overall and per-agent returns from evaluation runs.



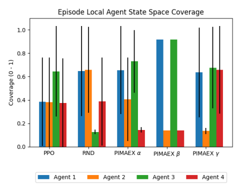
(a) Exploration state space coverage (b) Agent state space coverage
Figure 3: Final state space coverage for best training run of each agent category.

Differences in state space coverage are less pronounced than those observed in returns. While final exploration state space coverage varies slightly among methods (except for PPO), differences are

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS



(a) Exploration state space coverage

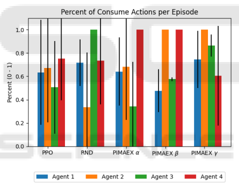


(b) Local agent state space coverage

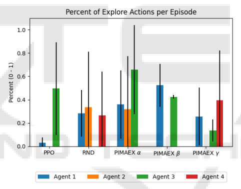
Figure 4: Per-episode exploration and local agent state space coverage from evaluation runs.

more evident when examining coverage within an episode. Here, *PIMAEX* α agents are the best explorers, despite not participating in other agents' intrinsic returns like the β and γ agents. This suggests that influence rewards combined with individual curiosity may suffice to enhance multi-agent exploration. Notably, *PIMAEX* β agents exhibit significantly reduced standard deviation in both local agent state space coverage within an episode and final agent state space coverage after training.

An interesting observation is that *PIMAEX* β agents appear to specialize in teams of two: agents 1 and 3 are the best explorers, while agents 2 and 4 explore less, especially within an episode.



(a) Consume actions per agent

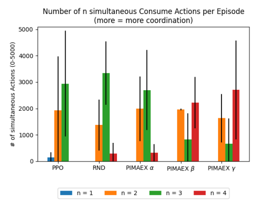


(b) Explore actions per agent

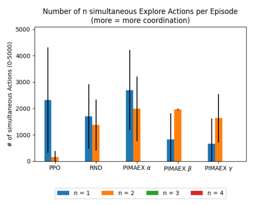
Figure 5: Per-episode percentage of consume and explore actions per agent.

Fig. 5 confirms these findings. It shows that *PIMAEX* β agents 2 and 4 predominantly consume and rarely explore. Again, *PIMAEX* α agents are the most active explorers, followed by RND and *PIMAEX* γ. This supports the hypothesis that participating in others' intrinsic returns does not necessarily promote more exploration, and that individual intrinsic returns, possibly combined with influence rewards, can drive multi-agent exploration.

Examining the number of simultaneous consume (left) or explore (right) actions in Fig. 6, none of the agent classes coordinate explore actions in teams larger than two, though they do so for consumption. When focusing on pairs of simultaneous explore actions, *PIMAEX* agents explore in teams of two for about one-third of the episode, closely followed by RND. Again, *PIMAEX* β agents display considerably less standard deviation than others.

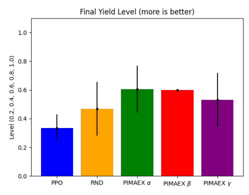


(a) Simultaneous consume actions

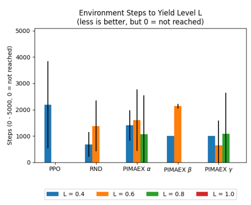


(b) Simultaneous explore actions

Figure 6: Number of simultaneous consume and explore actions per episode from evaluation runs.



(a) Final yield level



(b) Steps to reach yield level

Figure 7: Final yield level and steps to reach yield level from evaluation runs.

7 CONCLUSION

This work introduced two reward functions: the *PIMAEX* reward, a peer incentivization mechanism based on intrinsic curiosity and social influence, and a more generalized version usable by agents without intrinsic curiosity (though not evaluated here). The *PIMAEX-Communication* training algorithm, compatible with any actor-critic method, adopts a communication mechanism from (Jaques et al., 2018) and can be implemented easily atop existing algorithms. The *Consume/Explore* environment presented in this work is also a flexible tool for researching multi-agent reinforcement learning. Empirically, *PIMAEX-Communication* improves overall returns in the *Consume/Explore* task compared to baselines without social influence. Notably, *PIMAEX* β achieves the highest return, though the findings indicate that participating in others' intrinsic returns does not always yield more exploration. Interestingly, *PIMAEX* α, which relies only on social influence and individual curiosity, exhibits the strongest exploratory behavior, and *PIMAEX* β shows the most stable policies.

However, this work has limitations. It uses only small, feed-forward networks and evaluates *PIMAEX* with PPO on a single task and limited training time. Future work should explore larger or recurrent architectures, alternative actor-critic methods (e.g., IMPALA(Espeholt et al., 2018)), and more complex settings with larger state and action spaces or more agents. Addressing these limitations will help deter-

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS

mine the broader effectiveness of *PIMAEX* in multi-agent reinforcement learning.

ACKNOWLEDGEMENTS

This work is part of the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus. This paper was partly funded by the German Federal Ministry of Education and Research through the funding program "quantum technologies — from basic research to market" (contract number: 13N16196).

REFERENCES

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. (2018). Exploration by random network distillation.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. (2018). Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures.

Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2017). Counterfactual multi-agent policy gradients.

Fu, W., Yu, C., Xu, Z., Yang, J., and Wu, Y. (2022). Revisiting some common practices in cooperative multi-agent reinforcement learning.

Hoffman, M. W., Shahriari, B., Aslanides, J., Barth-Maron, G., Momchev, N., Sinopalnikov, D., Stańczyk, P., Ramos, S., Raichuk, A., Vincent, D., Hussenot, L., Dadashi, R., Dulac-Arnold, G., Orsini, M., Jacq, A., Ferret, J., Vieillard, N., Ghasemipour, S. K. S., Girgin, S., Pietquin, O., Behbahani, F., Norman, T., Abdolmaleki, A., Cassirer, A., Yang, F., Baumli, K., Henderson, S., Friesen, A., Haroun, R., Novikov, A., Colmenarejo, S. G., Cabi, S., Gulcehre, C., Paine, T. L., Srinivasan, S., Cowie, A., Wang, Z., Piot, B., and de Freitas, N. (2020). Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*.

Jaques, N., Lazaridou, A., Hughes, E., Gulcehre, C., Ortega, P. A., Strouse, D., Leibo, J. Z., and de Freitas, N. (2018). Social influence as intrinsic motivation for multi-agent deep reinforcement learning.

Leibo, J. Z., Zambaldi, V., Lanctot, M., Marecki, J., and Graepel, T. (2017). Multi-agent reinforcement learning in sequential social dilemmas.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments.

Ostrovski, G., Bellemare, M. G., Oord, A. v. d., and Munos, R. (2017). Count-based exploration with neural density models.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V. (2007). Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction.

Peng, P., Wen, Y., Yang, Y., Yuan, Q., Tang, Z., Long, H., and Wang, J. (2017). Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games.

Rashid, T., Samvelyan, M., de Witt, C. S., Farquhar, G., Foerster, J., and Whiteson, S. (2018). Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning.

Schmid, K., Belzner, L., and Linnhoff-Popien, C. (2021). Learning to penalize other learning agents. In *Proceedings of the Artificial Life Conference 2021*, volume 2021. MIT Press.

Schmidhuber, J. (1991). A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior on From Animals to Animats*, page 222–227, Cambridge, MA, USA. MIT Press.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms.

Stadie, B. C., Levine, S., and Abbeel, P. (2015). Incentivizing exploration in reinforcement learning with deep predictive models.

Sukhbaatar, S., Szlam, A., and Fergus, R. (2016). Learning multiagent communication with backpropagation.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. (2016). #exploration: A study of count-based exploration for deep reinforcement learning.

Wang, T., Wang, J., Wu, Y., and Zhang, C. (2019). Influence-based multi-agent exploration.

Yang, J., Li, A., Farajtabar, M., Sunehag, P., Hughes, E., and Zha, H. (2020). Learning to incentivize other learning agents.

SCIENCE AND TECHNOLOGY PUBLICATIONS

APPENDIX

Table 1: Environment hyperparameters and settings used by all experiments.

Parameter	Value
Number of agents (N)	4
Episode length in steps	5000
Reward per consumption (R)	1
Exploration failure penalty (P)	0
Production cycle time in steps (M)	10
Initial production yield level (C_{init})	1
Maximum production yield level (C_{max})	5
Initial supply (S_{init})	0
Maximum supply depot capacity (S_{max})	10
Exploration success threshold E	0.5
Num. of successful explore actions c_{max}	2000

Table 2: Common training hyperparameters and settings used by all experiments. All training runs were repeated with three seeds and results averaged.

Parameter	Value
Environment training steps	1e7
Optimiser	Adam
Learning rate	1e-4
Adam ε	1e-7
Max. gradient norm	0.5
Batch size	16
Unroll length	128
Num minibatches	4
Num epochs	4
Discount γ_E	0.999
GAE λ	0.95
Entropy cost	1e-3
PPO clipping ε	0.1
Num Actor Processes	16

Table 3: Training hyperparameters and settings used by all PPO+RND experiments. Lists of values indicate these were included in hyperparameter search, whereas all other values remain fixed in all training runs.

Parameter	Value
Intrinsic discount γ_I	0.99
Infinite time horizon for intrinsic return	True
Extrinsic and intrinsic reward coefficients	[(2.0, 1.0), (1.0, 0.5)]
Max. abs. intrinsic reward	[False, 1.0]
Separate neural network for intrinsic value	[False, True]
Proportion of experience used for training RND predictor	[0.25, 1.0]
RND observation normalisation initialisation environment steps	1e5

Table 4: Training hyperparameters and settings used by all PIMAEX-Communication experiments. Lists of values indicate these were included in hyperparameter search, whereas all other values remain fixed in all training runs.

Parameter	Value
Communication discount γ_C	0.99
Communication entropy cost	7.89e-4
Communication loss weight	[1.0, 0.0758]
Communication reward coefficients (for extrinsic, intrinsic, and *PIMAEX* rewards)	(0.0, 0.0, 2.752)
Policy influence measure	$[KL_D, PMI]$
Extrinsic and intrinsic reward coefficients	(1.0, 0.5)
Max. abs. intrinsic reward	False
Separate neural network for intrinsic value	False
Proportion of experience used for training RND predictor	0.25

Table 5: Hyperparameters of best-performing *PIMAEX* agents.

Parameter	*PIMAEX* α	*PIMAEX* β	*PIMAEX* γ
Communication loss weight	0.0758	1.0	0.0758
Policy influence measure	KL_D	*PMI*	-
Extrinsic/intrinsic coefficients for β and γ		(0.0, 1.0)	(0.0, 1.0)
α	1.0	0.0	0.00
β	0.0	1.0	0.00
γ	0.0	0.0	0.01

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS