Cooperation or coincidence ? Explaining cooperation in multi-agent reinforcement learning

Loïc Cordeiro Fonseca[1] [0009−0007−2697−9981]
Yannick Molinghen[1] [0009−0008−2757−3737]
Tom Lenaerts[1,2,3] [0000−0003−3645−1455]

[1] Machine Learning Group, Université Libre de Bruxelles, Brussels, Belgium
[2] AI Lab, Vrije Universiteit Brussel, Brussels, Belgium
[3] Center for Human-Compatible AI, UC Berkeley, USA

Abstract. Deep Reinforcement Learning has achieved important progress in complex environments but remain difficult to interpret due to the opacity of deep neural policies. This challenge becomes even more pronounced in cooperative Multi-Agent Reinforcement Learning where coordination between agents must be understood alongside individual decision-making. In this work, we investigate whether cooperation is explicitly encoded in agent policies or if it emerges as the by-product of selfish incentives. We propose a reward decomposition framework that categorizes reward components in cooperative or selfish categories to explain cooperative behaviours and apply this method in the Laser Learning Environment where agents heavily rely on each other to succeed. Our approach enables an analysis at two levels: locally, by identifying which incentives dominate specific actions, and globally, by tracking how priorities evolve during training. Overall, our experiments uncover explicit cooperation in key transitions while also exposing the persistence of selfish incentives.

Keywords: Reinforcement Learning · Multi-Agent · Explainability.

1 Introduction

In its early days, Reinforcement learning [Sutton and Barto, 2018, RL] relied on tabular methods where policies and value functions are stored explicitly in tables that provide a transparent and interpretable mapping from states and actions to expected returns. These approaches allowed researchers and practitioners to directly inspect, analyse, and understand the decision-making process of an agent. As RL scaled to high-dimensional and complex environments, the use of deep neural networks became necessary to approximate policies and value functions [Mnih et al., 2015; Schulman et al., 2017]. This shift to deep RL has led to remarkable breakthroughs with the caveat that the learned policies are encoded in large, distributed parameter spaces that are difficult to interpret, making it significantly more difficult to explain why an agent behaves the way it does.

In this context, Explainable RL [Puiutta and Veith, 2020; Milani et al., 2024, XRL] has emerged as a response to the growing opacity of deep RL policies with the aim to provide insights into the agents' learning or decision making. By improving interpretability at various levels, XRL increases the confidence that humans can put in the agents. Extending the pursuit of explainability to cooperative Multi-Agent RL [Albrecht et al., 2024, MARL] comes with new and significant challenges and gave birth to the field of explainable MARL [Zabounidis et al., 2023; S. Milani et al., 2022, XMARL].

Recent works in the field of XMARL have tackled multiple questions such as agent-wise credit assignment [Heuillet et al., 2022], division of labour among the agents [Kazhdan et al., 2020; Khlifi et al., 2023] or even the integration of other agents into existing XRL techniques [S. Milani et al., 2022]. One particularly interesting XMARL question that remains unanswered is whether the cooperation observed in trained agents is genuinely encoded in their policies or if it arises as an incidental by-product of optimization, a kind of "happy coincidence" rather than a deliberate, learned strategy. Understanding the difference is critical for both theoretical and practical reasons: true cooperative behaviour suggests robustness, adaptability, and generalization across scenarios, whereas accidental cooperation may fail under even slight environmental or task variations.

In this work, we leverage Reward Decomposition [Juozapaitis et al., 2019, RD] as an interpretability tool that disentangles the long-term objectives embedded in agents' policies and classifies them as cooperative or selfish. Our approach enables analyses at two complementary levels: locally, by clarifying which incentives drive specific state–action choices; and globally, by tracking how priorities evolve over the course of the training. By applying our method to the Laser Learning Environment – where success requires fine-grained coordination – we show that it reveals when cooperation is explicitly encoded in policies rather than merely emerging as a side-effect of selfish behaviour. Additionally, we explain why neural-network mixing methods such as QMIX are bad candidates in the scope of XMARL, and prove that they are theoretically unsuitable to RD. To the best of our knowledge, this is the first work to leverage RD to disentangle genuine cooperation from coincidental one, providing both methodological contribution to XMARL and insights in the dynamics of learned cooperative strategies. The code to reproduce our experiments can be found at https://github.com/yamoling/marl/releases/tag/bnaic-2025.

2 Background

2.1 Multi-Agent Reinforcement learning

In Reinforcement Learning [Sutton and Barto, 2018, RL], agents learn a policy by interacting with an environment and by receiving rewards or punishments according to their actions. The process of decision-making under uncertainty is modelled by a Markov Decision Processes [Bellman, 1957, MDP] as the tuple $\langle S, A, T, R \rangle$, where S is the finite set of states, A is the finite set of actions,

$T : S \times A \times S \to [0, 1]$ is the transition that gives the probability to land in state s' by taking action a in state s, and $R : S \times A \times S \to \mathbb{R}$ is the reward function.

Similarly, a Multi-agent Markov Decision Process [Boutilier, 1996, MMDP] is described as a tuple $\langle n, S, A, T, R \rangle$ where S, T and R are the same as for MDPs, n is the number of agents, and $A \equiv A_1 \times \cdots \times A_n$ is the set of joint actions with A_i being the set of actions of agent i.

In RL, the objective of the agent is to find the policy $\pi : S \to \Delta_A$ that maximizes the expected sum of discounted rewards $\mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 = s\right]$, where γ is a discount factor that weights the importance of future rewards in comparison to immediate ones. The action-value function under a given policy is $Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 = s, a_0 = a\right]$ and is referred to as Q-function. The output of a Q-function is referred to as Q-value.

2.2 Value function factorisation

Laurent et al. [2011] showed that multi-agent systems are susceptible to a non-stationarity problem, which Tuyls and Weiss [2012] also refer to as the multi-agent moving target problem. This arises because learning agents perceive the other learning agents as part of the environment. As a result, in a given state, an agent may perform the same action twice but observe different outcomes due to the actions of other agents. Claus and Boutilier [1998] have shown that naive implementations of Independent Q-Learning (IQL) were often unsuccessful, even for very simple tasks, partly because of this non-stationarity.

To tackle these challenges, Oliehoek et al. [2008] propose to factor out the value function, and Sunehag et al. [2018] introduce Value Decomposition Network (VDN), a MARL algorithm in which each agent i has its own utility function $Q^i : S \times A_i \to \mathbb{R}$. The authors then optimize the Q-network under the assumption that the joint Q-value is the sum of the agents' utility, i.e. $Q^{\text{VDN}} = \sum_{i=1}^{n} Q^i$. This factorization allows for decentralized execution thanks to the Individual Global Max property [Son et al., 2019, IGM], which ensures consistency in action selection between the centralized training and the decentralized execution. Rashid et al. [2018] extend the principle of Q-value factorization with QMIX, identifying that any monotonically increasing function respects the IGM property, thereby increasing the range of functions that can be used and therefore the representation capabilities of mixing functions.

2.3 Laser Learning Environment

The Laser Learning Environment [Molinghen et al., 2025, LLE] illustrated in Figure 1 is the multi-agent environment used throughout this work. LLE is a cooperative grid world where coordination is essential. Agents operate on a grid and can move in the four cardinal directions or remain stationary. The objective of the environment is to collect the gems and reach the designated exit tiles while avoids hazards such as holes and lasers.

Lasers are a central feature of LLE. Each laser is associated with a specific agent colour. An agent can block lasers of its own colour, but stepping into a laser

of a different colour results in the agent's death, a negative associated reward, and in the premature termination of the episode. This design enforces both *inter-agent dependence* – as agents must rely on each other to safely navigate the environment – and *perfect coordination*, since even small discrepancies in timing or movement can lead to failure.

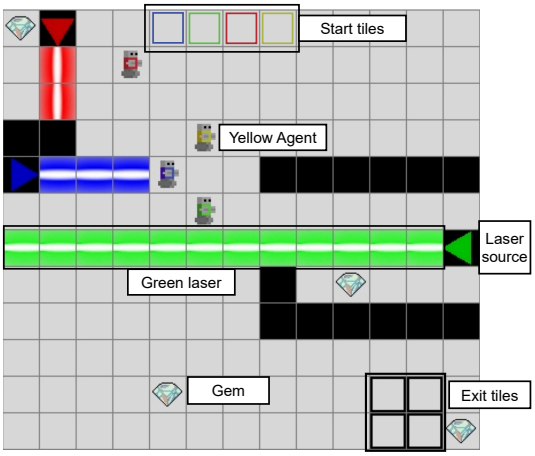


Fig. 1: An annotated representation of the Laser Learning Environment where the blue agent is blocking the blue laser for other agents to proceed south.

The standard observation that agents receive in LLE is a representation of the entire game board and an array of extra features. The game board representation is layered, meaning that each feature is one-hot encoded on the board in an individual layer.

3 Related works

We give an overview of related works in the field of XRL. We discuss the two techniques similar to ours and explain how they differ from our own approach.

3.1 Diagnostics for MARL training

Khlifi et al. [2023] introduce statistics driven metrics to help identify hidden behavioural trends. Building on top of the policy entropy [Abdallah, 2009] which quantifies how uncertain or deterministic an agent's policy is, the authors compute the agent update divergence. This metric measures how much agents policies change from one time-step to another during training by computing the KL divergence [Kullback and Leibler, 1951] of the policy entropy in two consecutive steps. Finally they quantify how often agents change actions during evaluation,

resulting in the distribution of actions they take over the course of training, informing analysts of how labour is divided among agents.

These metrics enable analysis of the agents joint strategy and by extension the identification of flaws in said strategy. However, they do not dive deeper to provide insights on the motivations of agents during training.

3.2 Reward decomposition

Reward decomposition [Russell and Zimdars, 2003; Juozapaitis et al., 2019, RD] is an approach to XRL that relies on the definition of a set of reward components $C = \{c_1, c_2, \ldots, c_n\}$ that contribute to the reward function. For instance, in the case of self driving car agents, components could relate to traffic code c_{code} and fuel consumption c_{fuel}. In that setting, traffic rule violations would yield a punishment in c_{code}, and rewards related to fuel consumption would be issued in c_{fuel}. In RD, the reward function is defined as the sum of the reward components as shown in Equation 1.

$$\mathbf{R}(s_t, a, s_{t+1}) = \sum_{c \in C} R_c(s_t, a_t, s_{t+1}) \tag{1}$$

In turn, it is possible to train value or action-value functions to estimate each of the individual components and then to sum these functions to retrieve the global action-value function, as shown in Equation 2.

$$\mathbf{Q}(s, a) = \sum_{c \in C} Q_c(s, a) \tag{2}$$

In essence, RD enhances the interpretability of the agent's policy by making the underlying objectives transparent and by enabling analysts to understand and diagnose how specific environmental cues influence agent behaviour.

Juozapaitis et al. [2019] also present Reward Difference Explanations (RDX) and Minimal Sufficient Explanations (MSX). These metrics allow them to quantify the impact of individual components and highlight the most impactful components for or against a specific decision. With that, they focus on individual decisions in a single-agent settings

In a recent work, Iturria-Rivera et al. [2024] use RD in a multi-agent setup and combine RD with Q-value factorisation in order to understand which component contributes the most or the least to the policy of their multi-agent system. However, the authors do not tackle the question of agent cooperation, which is at the centre of this work.

Both of these works limit their interpretations to a decision-making level and do not apply reward decomposition to a larger scope. This work also extends the technique to a global scope to get insights on policies and the learning process.

4 Highlighting cooperation via Reward Decomposition

In this section, we investigate how RD can be used to show the presence of cooperation in LLE. We first present our methodology, describe how we decompose the

reward function of LLE, explain our experimental setup and how we gathered the Q-values for our interpretations. Finally, we present the insights and explanations obtained from the reward decomposition with regard to cooperation.

4.1 Method

We work with the shaped version of LLE introduced by Molinghen and Lenaerts [2025a] that promotes laser-blocking behaviour and enables state-of-the-art multi-agent algorithms to complete the task. We decompose the reward signal of LLE into the five components illustrated in Table 1. We categorize the reward components as selfish or cooperative depending on the level of coordination required to acquire the reward and on the impact of collecting that reward on the other agents.

On the one hand, since collecting a gem or reaching an exit can generally be done single-handedly, these actions are categorised as selfish, with R_{exit} being particularly so, as an agent's exit from the environment prevents any further action and therefore any coordination. On the other hand, receiving a penalty for dying R_{death} ends the current episode, which affects all of the agents, R_{done} can only be achieved when all of the agents have reached an exit, and R_{pbrs} requires other agents to block the laser to collect the corresponding reward. As a result, the three latter components are categorized as cooperative.

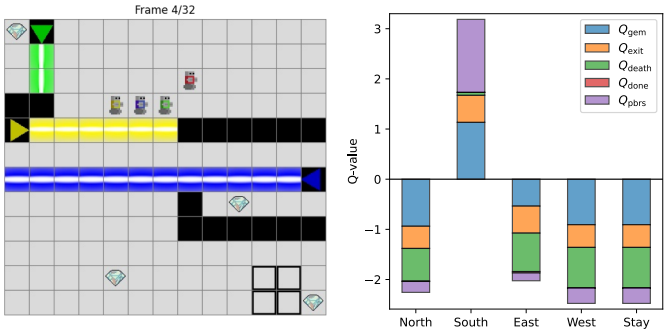
Table 1: LLE reward decomposition with their signal and classification.

Component	**Signal**	**Classification**
R_{gem}	$+1$ when a gem is picked up	Selfish
R_{exit}	$+1$ when an agent enters an exit tile	Selfish
R_{death}	-1 when an agent dies	Cooperative
R_{done}	$+1$ when agents have arrived	Cooperative
R_{pbrs}	$+1$ the first time each agent crosses each laser without dying (Potential-Based Reward Shaping)	Cooperative

We adapt VDN [Sunehag et al., 2018] and QMIX [Rashid et al., 2018] to accommodate RD by vectorizing the outputs of the individual Q^i-networks to size $|C|$, and then applying the mixing on a per-component basis. The corresponding equations are respectively shown in Equation 3 and Equation 4, where $f_{\theta(s)}$ is a monotonically increasing function.

$$\mathbf{Q}_c^{\text{VDN}}(s,a) = \sum_{i=1}^{N} Q_c^i(s,a) \tag{3}$$

$$\mathbf{Q}_c^{\text{QMIX}}(s,a) = f_{\theta(s)}(Q_c^1(s,a), \ldots, Q_c^n(s,a)) \tag{4}$$



(a) (b)

Fig. 2: (Left) Representation of the first LLE state under study for Feature Importance analysis. The yellow agent is about to block the yellow laser. (Right) Component-wise Q_c^{yellow}-values for the state represented on the left. The stacked bars account for the total Q^{yellow}-value.

4.2 Experimental Setup

We train cooperative agents on the map shown in Figure 2a where the yellow agent and the blue agent should go down the central part of the map, blocking their associated lasers in the process and allowing the green and the red agent to pass through towards the exit located in the bottom right corner of the grid.

We train agents with VDN and QMIX adapted for RD and collect the agents' best action Q^i across every reward component every 5000 time steps over the course of the training. Agents use an ϵ-greedy policy with ϵ decaying from 1 to 0.05 over 200k time steps, and the length of each episode is capped to $\lfloor \frac{width \times height}{2} \rceil = 78$ time steps, i.e. enough to discover the environment without collecting too many irrelevant transitions. Since this work focuses on explainability techniques and the interpretations that they provide, the other hyper-parameters and the Q-network architecture can be found in Appendix B.

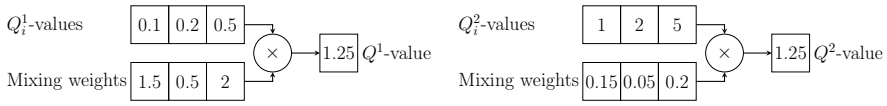
Note that we use the shaped version of LLE introduced in [Molinghen and Lenaerts, 2025b] because Molinghen et al. [2025] have shown in a previous work that it was a very difficult task that VDN and QMIX were unable to solve otherwise.

4.3 Results

We repeat our experiments with 12 different seeds and analyse the Q^i-values with regard to agents cooperation both on the feature importance and on the policy learning level. We first explain why QMIX results were inconclusive, and then focus on the results of our VDN experiments. Nonetheless, the policy learning level results using QMIX are shown in Appendix C.

Inapplicability of QMIX Our results with QMIX were inconclusive, both in terms of performance and in in terms of explainability (on the feature importance level and on the policy learning level). In this section, we give an intuition of why neural-network based mixing methods such as QMIX or QPLEX fail with RD and give a detailed proof in Appendix A.

Intuitively, consider the fact that mixing networks can end up in different local optima from one component to an other, thereby possibly scaling the agents' Q_i^c-values up or down accordingly. Since every mixing network is optimized against it own component, it is possible for the componentQ_i-values of agents to be of different orders of magnitude, which we illustrate in Figure 3.



(a) Q^1 computation (b) Q^2 computation

Fig. 3: Fictive weights for mixing networks for components 1 and 2. The joint action-value Q^c is 1.25 in both cases, although the individual agents' Q_i^c-values are of different orders of magnitude.

Even though each individual mixing network ensures the IGM property discussed in Section 2.2 at the component-level, the action selection as defined in Equation 2 is no longer consistent because it does not account for this potential scaling. We illustrate this with a fictional example in Table 2 where components 1 and 2 are respectively scaled by w_1 =0.1 and w_2 =1.0 by their mixing networks. The action selected by Equation 2 is action 2. However, if we scale these values by the weight of their respective mixing networks, the best action is action 3.

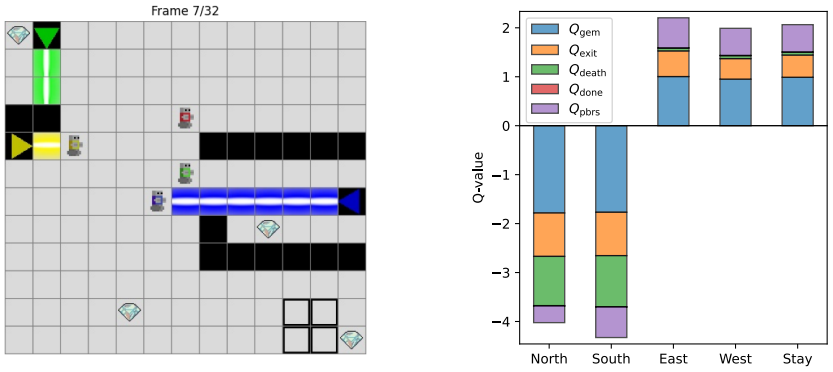
Table 2: Fictional Q_i^c-values of an agent trained with non-linear component-wise mixing networks. The orders of magnitude of the Q_i^c-values are very different from one component to the next, resulting in inconsistent total Q_i-values.

	Action 1	**Action 2**	**Action 3**
Component 1	**10.5**	10.7	9.0
Component 2	-1.0	-1.0	**0.2**
$\sum_{c \in C} Q_c^i(s,a)$	9.5	**9.7**	9.2
$\sum_{c \in C} w_c Q_c^i(s,a)$	0.05	0.07	**1.1**

To conclude, the scaling of Q_i^c occurs at training time (at the good will of the optimizer) but the agent is unaware of it at decision-making time, which *can* result in a good policy. That being said, independently of the quality of the

learned policy, this difference in scales has consequences on the explainability side. Since the decision-making has become inconsistent with the training, it is no longer possible to interpret Q_i^c-values across components at the decision-making level.

Feature Importance level By analysing the individual agents' Q^i-values during the transitions shown in Figure 2a, Figure 4a and Figure 5a, we gather insights on the long term reward components that motivate the agents' decisions.



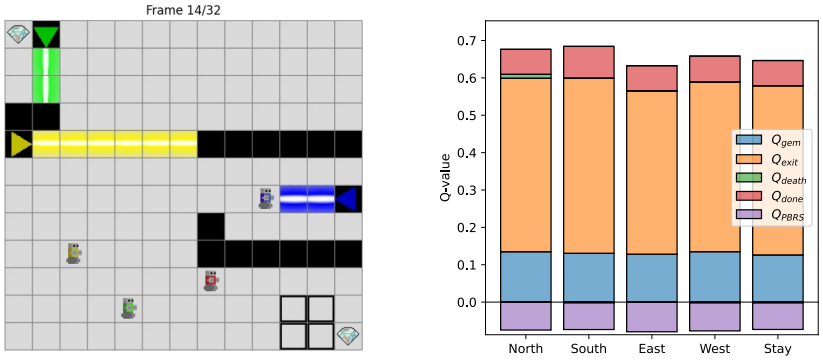
(a) (b)

Fig. 4: (Left) Representation of the second LLE state under study for Feature Importance analysis. (Right) Component-wise Q_c^{yellow}-values for the state represented on the left. The stacked bars account for the total Q^{yellow}-value.

Consider the state shown in Figure 2a where agent yellow should go south and block its laser to enable the other agents to pass and make progress in the collaborative task. Figure 2b illustrates the Q^{yellow}-values in this state by the end of the training. We can see that the most important incentive moving south is $Q_{\text{pbrs}}^{\text{yellow}}$ which is sensible since the agent will receive a reward of $+1$ when it enters this laser for the first time. Any other action is heavily dis-incentivized, with $Q_{\text{death}}^{\text{yellow}}$ accounting for a large part of the negative values, even though agent yellow is not at risk in any way. This clearly indicates *explicit coordination*, with agent yellow prioritizing a cooperative long term reward, and considering the risk of death of other agents as indicated by the negative Q^{yellow}-values.

This consideration is visible in Figure 4a as well, where actions north and south have a high negative and $Q_{\text{death}}^{\text{yellow}}$ accounting for a large part of it due to both agents green and red potentially being in range of the laser at the next step.

Meanwhile, if no agent is under threat by a laser, as shown in Figure 5a where all agents have already passed the blue laser, the incentive related to the death



(a) (b)

Fig. 5: (Left) Representation of the third LLE state under study for Feature Importance analysis.(Right) Component-wise Q_c^{blue}-values for the state represented on the left. The stacked bars account for the total Q^{blue}-value.

reward $Q_{\text{death}}^{\text{blue}}$, is negligible for all actions as shown in Figure 5b. We provide additional evidence of explicit cooperation from the point of view of other agents in Appendix D.

Policy learning level Figure 6 shows the relative importance of each reward component for agent yellow over the course of the training, averaged over the 12 repetitions. Each curve represents the prioritization order of reward components that drives the agent's decisions. Concretely, during training at each time-step, the optimal action's Q-values are collected (independently of the agent's effective action). Every 5000 time-step these collected Q-values are averaged and stored, representing the agent's prioritization with regards to objectives at that point during training. In the figure these values are normalized across categories for each time-step, so that the sum of all Q-values at a given time-step is 1.

We can see that Q_{death} positively dominates the others in the early stages of the training. This is expected since positive rewards are *sparse* in LLE, in particular before agents manage to coordinate effectively. Early on, the most reliable signal comes from the negative reward associated with dying. Consequently, the agents rapidly learn to take actions that take this signal into account, effectively avoiding deaths.

The associated positive value can be confusing and should not be interpreted as "by taking this action we get a positive death reward", but rather as "by taking this action we avoid a negative death reward". If stepping away from a laser results in an immediate null reward rather than the negative reward received by stepping in a laser, the result is relatively speaking positive. Thus the neural network learns to reinforce that behaviour resulting in a positive Q value. From

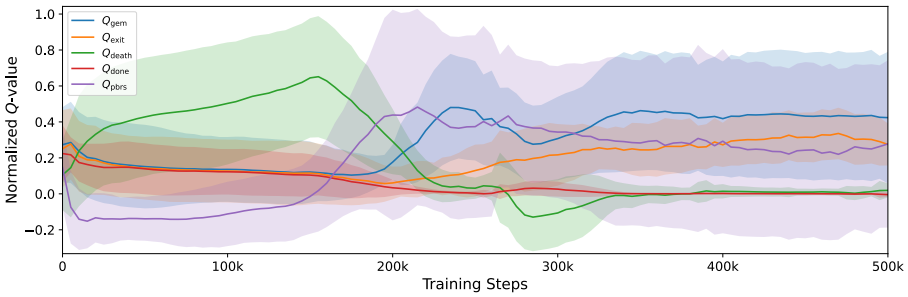


Fig. 6: Component importance on the course of the training for agent yellow. At each time-step the components Q_c^{yellow} are normalized to sum to 1 at each time step.

a long term point of view this action might eventually also result in an actual positive reward being received further reinforcing this notion. This is for instance illustrated in Figure 4b where Q_{gem} is evaluated negatively for actions North and South (although the gem component is always positive), while Q_{death} is slightly evaluated positively for the other actions (though the death component is always negative).

The initially negative Q_{pbrs} value shows that the -1 death penalty outweighs the $+1$ shaping reward, leading to an aversion toward the risk constituted by the traversal of lasers. Still, over time and thanks to the decrease of exploration, the shaping reward provides enough incentive for agents to attempt it, and the balance gradually shifts. This shift becomes visible around the 150k^{th} time step, when Q_{pbrs} starts to rise while Q_{death} declines, reflecting that agents learn to block lasers. Once Q_{pbrs} takes precedence over Q_{death}, agents also begin to prioritize Q_{gem} as they are now able to collect more of them.

As the training stabilizes, several long-term patterns tend to emerge: Q_{death} and Q_{done} converge to 0, indicating that death avoidance has been fully internalized and that all agents completing the task is not a strong driver. The narrow confidence interval and high value of Q_{exit} indicates that exiting the level is a consistent strong motivation across. Finally, Q_{gem} positively dominates the other components, meaning that agents come to prioritise the acquisition of gems over coordination.

5 Conclusion

In this work, we leveraged reward decomposition (RD) for explainable multi-agent reinforcement learning (XMARL) in coordination-critical environments and found evidence that cooperation is not a "happy coincidence" of a selfish policy but indeed encoded in the agents' individual policies. By decomposing the reward function into cooperative and selfish components, we provided a lens through which the motivations behind agents' behaviours can be analysed both

locally at the level of individual state–action transitions and globally over the course of training. Additionally, we intuitively showed and formally proved that for neural-network based mixing methods such as QMIX, there is a mismatch between the training and the exploitation of the policy under RD.

Our experiments in the Laser Learning Environment with Value Decomposition Network (VDN) demonstrated that RD offers concrete interpretability benefits. At the local feature importance level, we were able to identify explicit instances of cooperation such as agents prioritising laser-blocking even in the absence of selfish incentives. At the policy-learning level, we observed how agents initially relied on the avoidance of death as the dominant decision-making component, before progressively shifting towards more cooperative strategies. Importantly, our findings highlight the nuanced interplay between cooperative and selfish incentives: while cooperation emerges explicitly in key transitions, agents ultimately stabilise around policies that still reflect strong self-serving motivations such as collecting gems and exiting the level.

Our results suggest two important takeaways. First, reward decomposition provides a structured methodology to distinguish genuine cooperation from incidental coordination in MARL systems, contributing to the broader goal of building transparent and trustworthy AI. Second, our findings highlight the limitations of current cooperative training setups: even when coordination is required, agents may converge towards behaviours that prioritise selfish components once minimal cooperation is achieved.

Overall, this work demonstrates that RD is a valuable tool to interpret the opaque dynamics of cooperation in multi-agent reinforcement learning and to understand how cooperation emerges over the course of the learning.

Bibliography

Sherief Abdallah. Why global performance is a poor metric for verifying convergence of multi-agent learning, 2009. URL https://arxiv.org/abs/0904.2320.

Stefano V. Albrecht, Filippos Christianos, and Lukas Schäfer. *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024. URL https://www.marl-book.com.

Richard Bellman. *Dynamic programming*. Princeton Univ. Pr, 1957. ISBN 978-0-691-07951-6.

Craig Boutilier. Planning, Learning and Coordination in Multiagent Decision Processes. In *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*, Netherland, 1996. Morgan Kaufmann Publishers Inc.

Caroline Claus and Craig Boutilier. The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin, 1998.

Alexandre Heuillet, Fabien Couthouis, and Natalia Díaz-Rodríguez. Collective explainable ai: Explaining cooperative strategies and agent contribution in multiagent reinforcement learning with shapley values. *IEEE Computational Intelligence Magazine*, 17(1):59–71, 2022. https://doi.org/10.1109/MCI.2021.3129959.

Pedro Enrique Iturria-Rivera, Raimundas Gaigalas, Medhat Elsayed, Majid Bavand, Yigit Ozcan, and Melike Erol-Kantarci. Explainable multi-agent reinforcement learning for extended reality codec adaptation, 2024. URL http://arxiv.org/abs/2411.14264.

Zoe Juozapaitis, Anurag Koul, Alan Fern, Martin Erwig, and Finale Doshi-Velez. Explainable reinforcement learning via reward decomposition. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence Workshop on Explainable Artificial Intelligence*, 2019.

Dmitry Kazhdan, Zohreh Shams, and Pietro Lio. Marleme: A multi-agent reinforcement learning model extraction library. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

Wiem Khlifi, Siddarth Singh, Omayma Mahjoub, Ruan de Kock, Abidine Vall, Rihab Gorsane, and Arnu Pretorius. On diagnostics for understanding agent training behaviour in cooperative marl, 2023. URL https://arxiv.org/abs/2312.08468.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

Guillaume J. Laurent, Laëtitia Matignon, and N. Le Fort-Piat. The world of independent learners is not markovian. *International Journal of Knowledge-based and Intelligent Engineering Systems*, 15(1):55–64, March 2011. ISSN 18758827, 13272314. https://doi.org/10.3233/KES-2010-0206. URL https://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/KES-2010-0206.

Stephanie Milani, Nicholay Topin, Manuela Veloso, and Fei Fang. Explainable reinforcement learning: A survey and comparative review. *ACM Comput. Surv.*, 56(7), 4 2024. ISSN 0360-0300. https://doi.org/10.1145/3616864. URL https://doi.org/10.1145/3616864.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Yannick Molinghen and Tom Lenaerts. Zero-incentive dynamics: a look at reward sparsity through the lens of unrewarded subgoals, 2025a. URL https://arxiv.org/abs/2507.01470.

Yannick Molinghen and Tom Lenaerts. Zero-incentive dynamics: a look at reward sparsity through the lens of unrewarded subgoals, 2025b. URL http://arxiv.org/abs/2507.01470.

Yannick Molinghen, Raphaël Avalos, Mark Van Achter, Ann Nowé, and Tom Lenaerts. Laser learning environment: A new environment for coordination-critical multi-agent tasks. In Frans A. Oliehoek, Manon Kok, and Sicco Verwer, editors, *Artificial Intelligence and Machine Learning, BNAIC/Benelearn Conference Proceedings*, volume 2187 of *Communications in Computer and Information Sciences*, pages 135–154, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-74650-5.

Frans A Oliehoek, Matthijs T J Spaan, and Shimon Whiteson. Exploiting Locality of Interaction in Factored Dec-POMDPs. *Int. Joint Conf. on Autonomous Agents and Multi-Agent Systems*, pages 517–524, 2008. URL http://hdl.handle.net/10993/11029.

Erika Puiutta and Eric M. S. P. Veith. Explainable reinforcement learning: A survey. *CoRR*, abs/2005.06247, 2020. URL https://arxiv.org/abs/2005.06247.

Tabish Rashid, Mikayel Samvelyan, and Christian Schroeder. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of Machine Learning Research*, 2018. arXiv: 1803.11485v2.

Stuart Russell and Andrew Zimdars. Q-decomposition for reinforcement learning agents. In *Proceedings of the Twentieth International Conference on Machine Learning*, volume 2, pages 656–663, 01 2003.

and Nicholay Topin S. Milani, Z. Zhang, Zheyuan Ryan Shi, Charles Kamhoua, Evangelos E. Papalexakis, and Fei Fang. Maviper: Learning decision tree policies for interpretable multi-agent reinforcement learning, 2022. URL https://arxiv.org/abs/2205.12449.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv*, pages 1–12, 2017. ISSN 23318422. arXiv: 1707.06347.

Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning, May 2019. URL http://arxiv.org/abs/1905.05408. arXiv:1905.05408 [cs, stat].

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z.

Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, 3:2085–2087, 2018. ISSN 15582914.

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction*. Adaptive computation and machine learning series. The MIT Press, Cambridge, Massachusetts, second edition edition, 2018. ISBN 978-0-262-03924-6.

Karl Tuyls and Gerhard Weiss. Multiagent Learning: Basics, Challenges, and Prospects. *AI Magazine*, 33(3):41, September 2012. ISSN 2371-9621, 0738-4602. https://doi.org/10.1609/aimag.v33i3.2426. URL https://ojs.aaai.org/index.php/aimagazine/article/view/2426.

Renos Zabounidis, Joseph Campbell, Simon Stepputtis, Dana Hughes, and Katia Sycara. Concept learning for interpretable multi-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2302.12232.

A Mismatch between QMIX training and decision-making

We show in this section that there can be a mismatch between the policy training and the policy application when mixing networks such as QMIX or QPLEX are used due to the possible discrepancy of scaling across components.

Theorem 1. *Let $C = \{c_1, c_2, \dots\}$ be reward components, $A = \{1, 2, \dots\}$ be the action state, $Q_i^c(s, a)$ the state–action value of agent i for component c in state s, and suppose a mixing network for each component c that produces a joint component value Q^c and guarantees the component-wise IGM property, i.e. Q_i^c correctly ranks actions for agent i within component c.*

Let the state-action value of agent i be computed by summing $Q_i^c(s, a)$ across components, as stated by Equation 2.
If an action $a_i^\star$ is maximal across all components, i.e.

$$Q_i^c(s, a_i^\star) \geq Q_i^c(s, a) \quad \forall c \in C, \forall a \in A,$$

Then $a_i^\star$ is the optimal action.
Similarly, if a_i^- is minimal across all components, i.e.

$$Q_i^c(s, a_i^-) \leq Q_i^c(s, a) \quad \forall c \in C, \forall a \in A$$

Then a_i^- is the worst action.

Proof. We note f_{θ_c} the mixing function of component c (e.g. QMIX) and A_i the action space of agent i.
For the agents' action selection to be consistent with the mixing weights, we must ensure that for any agent i and for a fixed action selection of the other agents

$$\arg\max_{a_i \in A_i} \sum_{c \in C} f_{\theta_c}(Q_i^c(s, a_i), \cdot) = \arg\max_{a \in A_i} \sum_{c \in C} Q_i^c(s, a) \tag{5}$$

Where the right-hand-side is the action selection according to Equation 2.
We identify three cases: when action a is maximal across every component, when a is minimal across every component, and the general case.

Case 1: $a^\star$ is maximal across every component. We know that

$$f_{\theta_c}\left(Q_i^c(s, a_i^\star), \cdot\right) \geq f_{\theta_c}\left(Q_i^c(s, a_i), \cdot\right) \quad \forall a_i \in A_i$$

thanks to the IGM property. Therefore, $\sum_{c \in C} Q_i^c(s, a^\star)$ is also maximal.

Case 2: a^- is minimal across every component. Similarly to case 1, we know that

$$f_{\theta_c}\left(Q_i^c(s, a_i^-), \cdot\right) \leq f_{\theta_c}\left(Q_i^c(s, a_i), \cdot\right) \quad \forall a_i \in A_i$$

thanks to the IGM property. Therefore, $\sum_{c \in C} Q_i^c(s, a^\star)$ is also minimal.

Case 3: general case Let us illustrate with a counter-example that illustrates that Equation 5 does not always hold true. We consider two components c_1, c_2, three actions a_1, a_2, a_3, a mixing function f_{θ_c} that is a dot product with weights $[1.0, 0.1]$ (respectively noted w_1 and w_2) for the considered agent, and the Q_i^c-values shown in Table 3 that result in the selection of action 2 as the best action.

Table 3: Q_i^c-values of a agent i. The optimal action is action 2.

	a_1	a_2	a_3
c_1	-1.0	-1.0	**0.2**
c_2	**10.5**	10.7	9.0
$\sum_{c \in C} Q_c^i(s,a)$	9.5	**9.7**	9.2

When we adjust each component by its weight, we obtain Table 4 where the optimal scaled action is a_3.

Table 4: Q_i^c-values of a agent i scaled by their respective weights $w_1 = 1.0$ and $w_2 = 0.1$. The optimal action is action 3.

	a_1	a_2	a_3
$c_1 \times w_1$	-1.0	-1.0	**0.2**
$c_2 \times w_2$	**1.05**	1.07	0.9
$\sum_{c \in C} w_c Q_c^i(s,a)$	0.05	0.07	**1.1**

Equation 5 does not hold since the left-hand side yields a_3 and the right-hand side yields a_2. □

B Agent Hyperparameters

Table 5 shows the hyper-parameters used when training the DQN agents used in our experiments in LLE. These were of application independently of the Level, mixer and use of PBRS.

C Policy learning level results with QMIX

As is shown on Figure 7, the black-box mixing from QMIX results in inconclusive decompositions of Q-values.

Table 5: The hyper-parameters used across all experiments for training agents.

Hyperparameter	**Value**	**Comment**
Memory size	50000	Transitions
Batch size	64	Transitions
Train intervals	5	Time-steps
Optimizer	Adam	
α	5×10^{-4}	Learning rate - Both Q-network and mixer
Grad norm clipping	10	Both Q-network and mixer
γ	0.95	Discount factor
τ	0.01	Soft update rate
ϵ_{start}	1	
ϵ_{min}	0.05	
$\epsilon_{annealing}$	200000	Linearly annealed over time

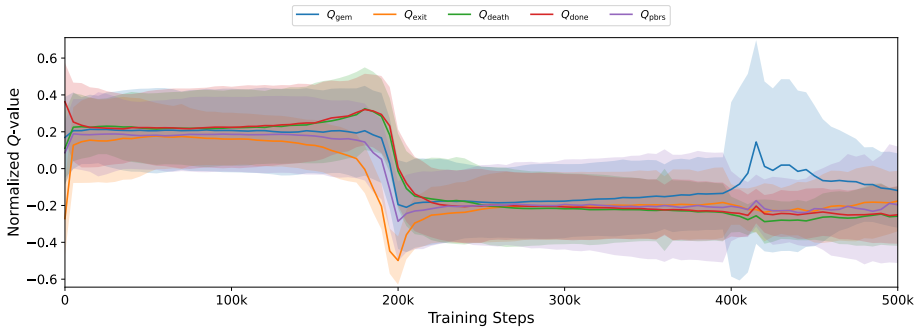


Fig. 7: Component importance on the course of the training for agent yellow with QMIX. At each time-step the components Q_c^{yellow} are normalized to sum to 1 at each time step.

D Complementary results of state-action RD

Agent blue's motivation (Figure 8) to step through the blocked laser, and to later block its laser in Frame 6 (left) is mostly motivated by Q_{PBRS}. Although there also happens to be a gem in the southern part which would be enough to send the agent southward. However, in frame 7 (right), agent blue goes eastward in large part due to Q_{gem} but also Q_{exit}. Among the five available actions, it also has the highest Q-value, albeit minimal, contribution from the PBRS objective. Picking the gem in the east, results in the laser being blocked longer (and twice considering the agent's way back), leaving more time to other agents to cross that laser which enables more agents to exit the level. The initial blocking of the laser is *explicit coordination*, motivated by Q_{PBRS} mainly. However, the coordination resulting in more agents traversing that laser is the result of a mix between cooperative objectives—more agents exiting, leading to a higher Q_{done} value—and a *self-serving policy*—picking up the gem.

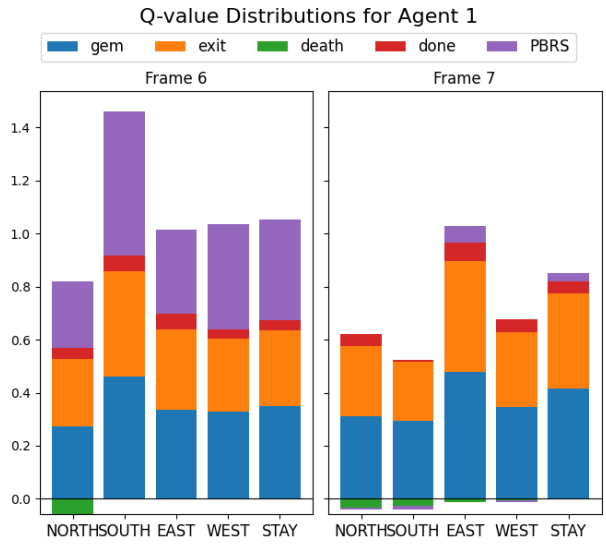


Fig. 8: Decomposed Q-values of agent blue (1) in frames 6 (left) and 7 (right)

In Frame 4, agent green has one of the clearer cases of PBRS' effect as shown in Figure 9. Most of their Q-values sends them northward, but a noticeably higher Q_{PBRS} value makes them go southward. This stronger Q_{PBRS} was the primary motivator for the southward action and is a clear indication of *explicit cooperation*. Agent red primarily goes southward in big parts because of Q_{PBRS} in frames 5 to 7.

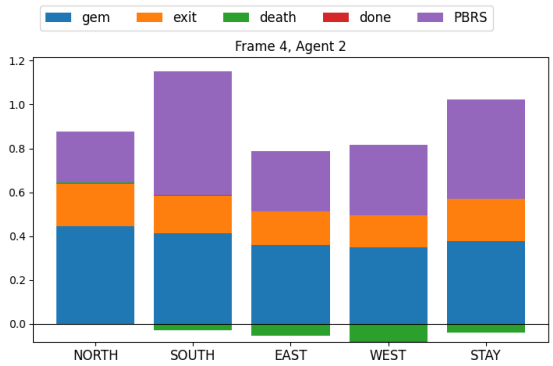


Fig. 9: Decomposed Q-values of agent green (2) at frame 4