1 Local Intrinsic Motivation

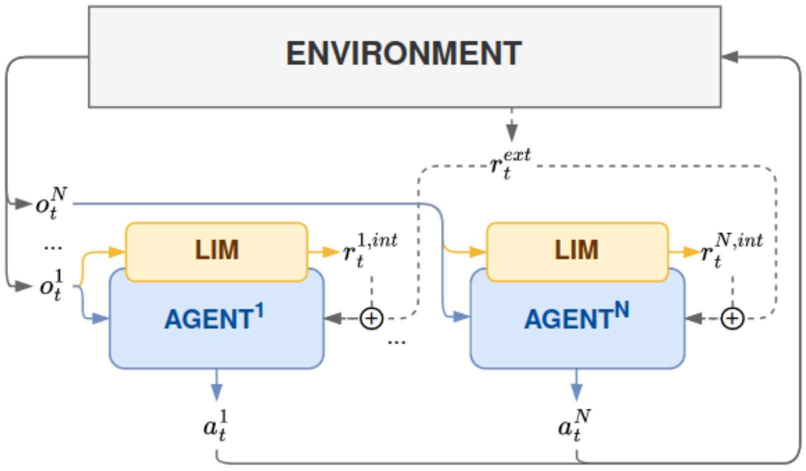

Figure 1: Architecture for local intrinsic motivation (LIM). Each agent has its own module for computing an intrinsic reward based on its local observation.

2 Hyperparameters

Hyperparameter	Algorithm	
	JIM	LIM
Intrinsic reward weight β	1	1
Encoding dim $D_{\phi/\psi}$	64	32
Hidden dim D_{hidden}	128	64
Scaling factor α	0.5	
Intrinsic reward learning rate α_{int}	0.0001	

Table 1: Hyperparameters used in the `rel_overgen` environment.

Hyperparameter	Algorithm	
	JIM	LIM
Intrinsic reward weight β	1	1
Encoding dim $D_{\phi/\psi}$	64	32
Hidden dim D_{hidden}	128	64
Scaling factor α	0.5	
Intrinsic reward learning rate α_{int}	0.0001	

Table 2: Hyperparameters used in the cooperative box pushing scenario.

Hyperparameter	Algorithm			
	JIM	LIM	JIM-LLEC	JIM-EEC
Intrinsic reward weight β	1, **2**, 4	1, **4**, 8	1, **3**	**0.1**, 1
Encoding dim $D_{\phi/\psi}$	64	36	64	64
Hidden dim D_{hidden}	512	256	512	512
Scaling factor α	0.5			
Intrinsic reward learning rate α_{int}	0.0001			

Table 3: Hyperparameters used in the coordinated placement scenario.

We present a list of hyperparameters used in each task presented in the paper. We only list hyperparameters specific to the intrinsic reward module, as for QMIX we use the default hyperparameters described in the original paper [22]. Here, we detail the different parameters presented:

- Intrinsic reward weight β: weight of the intrinsic reward r_i against the extrinsic reward r_e in the reward given to agents: $r_t = r_t^e + \beta r_t^{int}$.
- Encoding dimension $D_{\phi/\psi}$: dimension of the output of the embedding networks ϕ and ψ used in N_{LLEC} and N_{EEC} respectively (see Section 3).
- Hidden dimension D_{hidden}: dimension of the hidden layers in the embedding network ϕ and ψ used in N_{LLEC} and N_{EEC} respectively.
- Scaling factor α: parameter used in the definition of N_{LLEC} (see Eq. (6)). It controls how much novelty gain we want agents to find between each step.
- Intrinsic reward learning rate α_{int}: learning rate used for training the intrinsic reward module.

3 Details on our scenarios in the Multi-agent Particle Environment

3.1 Cooperative box pushing

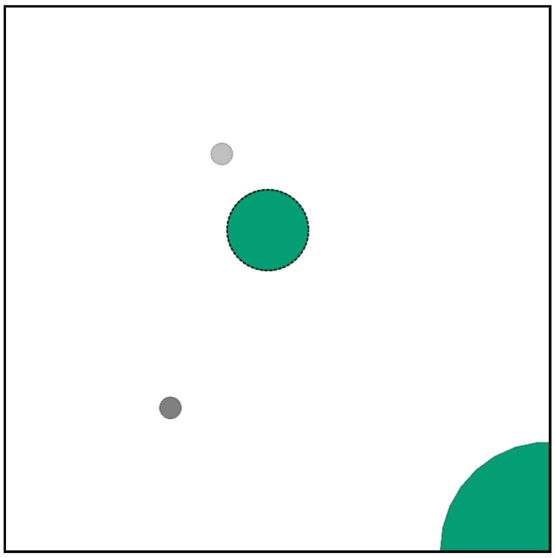

Figure 2: Cooperative box pushing task, agents are the small grey circles, the green circle in the middle is an object to deliver to the landmark in the bottom right corner.

The cooperative box pushing task requires pushing a round object and placing it on top of a landmark. The environment is a 2x2 meters area with walls on the sides that block entities from moving away. At each time step, an agent gets as observation:

- its personal data: its position and velocity $\text{pos}_{\text{self},x}$, $\text{pos}_{\text{self},y}$, $\text{vel}_{\text{self},x}$, $\text{vel}_{\text{self},y}$,
- data about the other agent: a boolean indicating if the other agent is visible or not, the relative position, and the velocity of this agent $\text{is\_visible}_{\text{agent}}$, $\text{dist}_{\text{agent},x}$, $\text{dist}_{\text{agent},y}$, $\text{vel}_{\text{agent},x}$, $\text{vel}_{\text{agent},y}$,
- data about the object: a boolean indicating if the object is visible or not, the relative position, and the velocity of this object: $\text{is\_visible}_{\text{object}}$, $\text{dist}_{\text{object},x}$, $\text{dist}_{\text{object},y}$, $\text{vel}_{\text{object},x}$, $\text{vel}_{\text{object},y}$,
- and data about the landmark: a boolean indicating if the landmark is visible or not and the number of the corner it is located into (from 1 to 4): $\text{is\_visible}_{\text{landmark}}$, $\text{corner}_{\text{landmark}}$.

Thus, the observation is a vector of dimension 16 containing this information. Relative positions of other entities (agent or object) are actually the distance to the agent, normalized by their range of observation, i.e.,

$$\text{dist}_{\text{agent},x} = \frac{\text{pos}_{\text{agent},x} - \text{pos}_{\text{self},x}}{\text{obs\_range}}.$$

The environment can be either fully observable or partially observable. In the latter case, agents have a range of observation of 60 centimeters around them. When agents or objects are outside this range, their relative position is masked with ones and their velocity with zeros. When the landmark is outside the range of observation, the corner number is masked with zero. In the fully observable case, the observation range is set to 2 meters and 83 centimeters, i.e., the largest distance possible to have in our 2x2 meters area.

The reward function is very sparse. At each time step, agents receive a penalty of 0.1, plus a penalty of 2 if there is a collision between agents. If the task is completed, i.e., the center of the object is placed in the area of the landmark, the agents get a reward of 100 and the episode ends.

At the start of each episode, the positions of all entities in the environment are randomly set: the agents and the object are randomly placed inside the environment, and the landmark is placed in one of the four corners of the map.

3.2 Coordinated placement

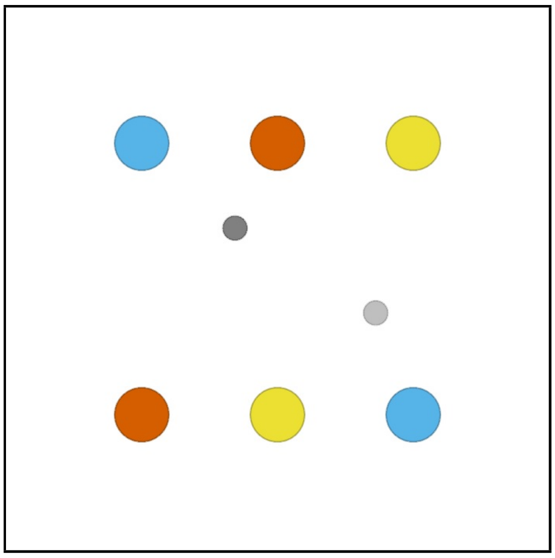

Figure 3: Coordinated placement task, agents are the small grey circles, the colored circles represent landmarks the agents have to navigate on to gain rewards.

The coordinated placement task requires navigating on top of landmarks and choosing the right landmark colors in order to maximize the obtained reward. The environment is a 2x2 meters area with walls on the sides that block entities from moving away. At each time step, an agent gets as observation:

- its personal data: its postion and velocity $\text{pos}_{\text{self},x}$, $\text{pos}_{\text{self},y}$, $\text{vel}_{\text{self},x}$, $\text{vel}_{\text{self},y}$,
- data about the other agent: a boolean indicating if the other agent is visible or not, the relative position, and the velocity of this agent $\text{is\_visible}_{\text{agent}}$, $\text{dist}_{\text{agent},x}$, $\text{dist}_{\text{agent},y}$, $\text{vel}_{\text{agent},x}$, $\text{vel}_{\text{agent},y}$,,
- for each landmark in the environment: a boolean indicating if the landmark is visible or not, the relative position of this landmark, and its color as a one-hot encoding: $\text{is\_visible}_{\text{landmark}}$, $\text{dist}_{\text{landmark},x}$, $\text{dist}_{\text{landmark},y}$, is_red, is_blue, is_yellow.

Thus, the observation is a vector of dimension 43 containing this information. Relative positions of other entities (agent or landmarks) are actually the distance to the agent, normalized by their range of observation, i.e.,

$$\text{dist}_{\text{agent},x} = \frac{\text{pos}_{\text{agent},x} - \text{pos}_{\text{self},x}}{\text{obs\_range}}.$$

This scenario is partially observable. Agents have a range of observation of 60 centimeters around them. When agents or objects are outside this range, their relative position is masked with ones and their velocity with zeros. For landmarks outside of the observation range, the color is masked with zeros.

The reward given at each time step depends only on which landmark has an agent placed on top:

- if two agents are placed on top of red landmarks, $r_t^e = 10$,
- if two agents are placed on top of blue landmarks, $r_t^e = 2$,
- if two agents are placed on top of yellow landmarks, $r_t^e = 1$,
- if only one agent is placed on top of either a blue or yellow landmark, $r_t^e = 0.5$,
- else, $r_t^e = 0$.

At the start of each episode, agents are placed randomly on the horizontal line in the middle of the map, i.e., $\text{pos} = (x = \text{uniform}(-1, 1), y = 0)$. The landmarks are always in the same positions, with the colors in the same order, as displayed in Figure 3.

4 Execution times

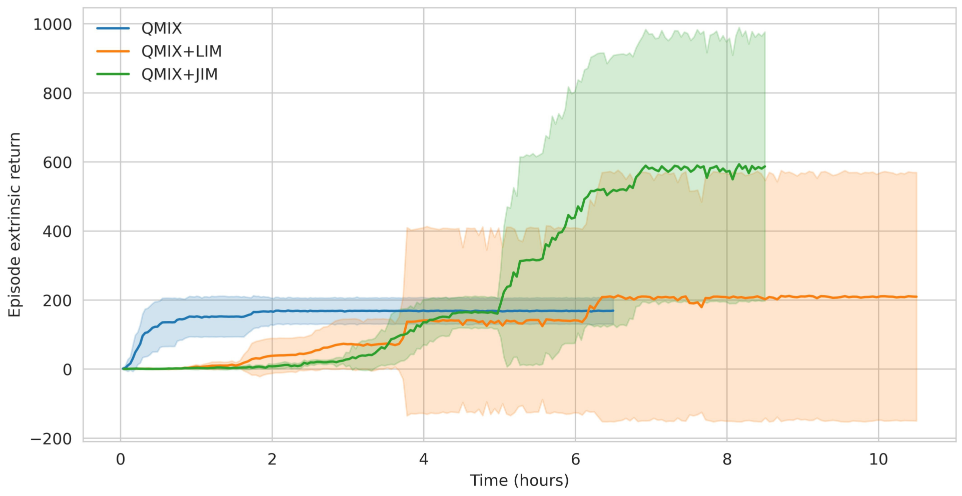

Figure 4: Training curves of QMIX, QMIX+LIM and QMIX+JIM in the coordinated placement task with execution time on the x-axis. QMIX takes on average 6.5 hours to train during 1e7 steps, while JIM takes 8.5 hours and LIM 10.5 hours.