MITIGATING REPRESENTATION DRIFT IN MARL WITH TRUST-REGION AUTOENCODERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-based multi-agent reinforcement learning (MARL) suffers from poor sample efficiency, limiting its practicality in real-world systems. Representation learning with auxiliary tasks can enhance efficiency; however, existing methods, including contrastive learning, often require the careful design of a similarity function and increase architectural complexity. In contrast, reconstruction-based methods that utilize autoencoders are simple and effective for representation learning, yet remain underexplored in MARL. We revisit this direction and identify unstable representation updates (*representation drift*) as a key challenge that limits the sample efficiency and stability in MARL. To address the challenge of representation drift, we propose the Multi-agent Trust Region Variational Autoencoder (MA-TRVAE), which stabilizes latent representations by regularizing updates within a trust region. Combined with Multi-agent Policy Proximal Optimization (MAPPO), MA-TRVAE improves its sample efficiency and stability in vision-based multi-agent control tasks. Experiments demonstrate that this simple approach is competitive with state-of-the-art MARL methods, while being more computationally efficient. Furthermore, we show that our method can scale up to more agents with only slight performance degradation.

1 INTRODUCTION

Recent advances in multi-agent reinforcement learning (MARL) have demonstrated its potential in sequential decision-making domains such as autonomous driving (Dinneweth et al., 2022), robotics (Orr & Dutta, 2023), and large-scale control (Ma et al., 2024), where multiple agents must learn to coordinate from partial and local observations. Nevertheless, MARL methods remain notoriously sample-inefficient, often requiring millions of interactions with the environment before converging to effective policies. This issue becomes more pronounced in vision-based settings, making learning slower, less scalable, and far less practical for real-world systems, where cameras are a convenient and inexpensive way to perceive the environment.

The sample efficiency of reinforcement learning with visual input has been extensively studied for single-agent systems. The most popular strategy is to learn a lower-dimensional representation of the high-dimensional input. This approach rests on the hypothesis that *learning a policy with a semantically meaningful low-dimensional representation of the visual input is significantly more sample efficient.* Different ways of obtaining such representations have been explored, including but not limited to (i) reconstruction-based representation learning (Yarats et al., 2021), (ii) contrastive learning (Laskin et al., 2020; Stooke et al., 2021), (iii) world models (Ha & Schmidhuber, 2018b; Hafner et al., 2019a; 2021; 2025), and (iv) pretrained encoders (Stooke et al., 2021; Nair et al., 2022). We discuss these works in more detail in the appendix A.

In the MARL literature, Song et al. (2023) utilizes contrastive learning and an attention module (Vaswani et al., 2017) to learn a representation that can capture temporal and agent-level information, while Feng et al. (2025) introduces a transition-informed framework to learn an attention-based world model that captures the dynamics of the multi-agent system. While these works, which we as well discuss in more detail in appendix A, improve sample efficiency in vision-based MARL, they rely on complex contrastive objectives, attention modules with quadratic input-length scaling, or learned dynamics models. We instead ask whether a simpler and more scalable approach is possible. In particular, we consider reconstruction-based representation learning, which avoids designing

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

complex similarity functions, scales linearly with image size, and does not require modeling partially observable multi-agent dynamics. Despite these advantages, this direction remains underexplored in MARL.

Following single-agent RL work (Yarats et al., 2021), we integrate *variational autoencoders* (VAEs) (Kingma & Welling, 2022) with *multi-agent proximal policy optimization* (MAPPO) (Yu et al., 2022) and find that weaker latent space regularization improves task performance in MARL. However, overly weak regularization can destabilize representation learning via *representation drift*: the same observation may map to substantially different latents across encoder updates, causing the policy to perceive it inconsistently and reducing sample efficiency. This instability is amplified in MARL due to diverse agent observations.

To address representation drift, we adapt the trust-region principle from policy optimization (Schulman et al., 2015; 2017b) to representation learning. To that end, we propose Multi-agent Trust Region VAE (MA-TRVAE), which regularizes updates to keep new representations close to previous ones. When combined with MAPPO, MA-TRVAE improves sample efficiency and stability across multiple vision-based control tasks while being more training-efficient than state-of-the-art methods. We then scale our methods to up to 7 agents and observe only a modest decline in performance, while other methods degrade significantly.

In summary, our main contributions are (i) we study reconstruction-based representation learning in MARL and identify representation drift as a challenge for MARL, (ii) we propose MA-TRVAE, a novel framework that adapts trust region principle to stabilize representation learning for MARL training, (iii) we demonstrate that MA-TRVAE improves sample efficiency and stability for MAPPO, outperforming baselines for the final return, while being more training efficient, and (iv) we show that MA-TRVAE scales effectively to more agents, maintaining strong performance while baselines degrade more rapidly. [1]

2 Problem Setting and Background

This section defines the problem setting and provides the required background on MAPPO, which we use as our baseline algorithm, and representation learning.

2.1 Problem Setting

We consider a cooperative MARL setting with partial observability for each agent, which we model as a decentralized partially observable Markov decision process (Dec-POMDP) (Oliehoek & Amato, 2016). A Dec-POMDP is defined as $\langle \mathcal{N}, \boldsymbol{\mathcal{O}}, \boldsymbol{\mathcal{A}}, R, P, \gamma \rangle$, where $\mathcal{N} = \{1, ..., N\}$ is the finite set of N agents, $\boldsymbol{\mathcal{O}} = \prod_{i=1}^{N} \mathcal{O}^i$ is the joint observation space, which is the Cartesian product of the local observation spaces $\boldsymbol{\mathcal{O}}^i \subseteq \mathbb{R}^{H \times W \times C}$ (since we consider vision-based observation, where H, W, and C are height, width, and number of channels of the image), $\boldsymbol{\mathcal{A}} = \prod_{i=1}^{N} \boldsymbol{\mathcal{A}}^i$ is the joint action space, composed of the local action spaces $\boldsymbol{\mathcal{A}}^i \subseteq \mathbb{R}^b$, $R : \boldsymbol{\mathcal{O}} \times \boldsymbol{\mathcal{A}} \to \mathbb{R}$ is the joint reward function, $P : \boldsymbol{\mathcal{O}} \times \boldsymbol{\mathcal{A}} \times \boldsymbol{\mathcal{O}} \to [0, 1]$ is the state transition probability function, and $\gamma \in [0, 1)$ is the discount factor. At each time step $t \in \mathbb{N}$, each agent receives a local observation $\boldsymbol{o}_t^i \in \boldsymbol{\mathcal{O}}^i$ and takes an action a_t^i according to its policy $\pi^i : \boldsymbol{\mathcal{O}}^i \to \boldsymbol{\mathcal{A}}^i$. The next set of observations $\boldsymbol{o}_{t+1}$ is updated based on the transition probability function P, and the entire team receives a joint reward $R(\boldsymbol{o}_t, \boldsymbol{a}_t)$. The goal is to maximize the expected cumulative joint reward over a finite or infinite number of steps,

$$\max_{\boldsymbol{\pi}} \mathbb{E}_{\boldsymbol{\pi}} \left[\sum_{t=0}^{T} \gamma^t R(\boldsymbol{o}_t, \boldsymbol{a}_t) \right]. \tag{1}$$

2.2 MAPPO

While in principle, agents have only access to their local actions and observations, MAPPO, which we use as the base MARL algorithm in this work, as many other MARL algorithms, relies on centralized training with decentralized execution (CTDE). With CTDE, agents have access to the global state and other agents' actions during training but use only local observations during execution.

[1] See https://sites.google.com/view/stablerepresentation for videos of the experiments and code.

MAPPO is an extension of *proximal policy optimization* (PPO) to MARL. MAPPO updates the parameters using the aggregated trajectories of all agents collected from the current policy. At iteration $k+1$, the policy parameters θ_{k+1} are optimized by maximizing the clipped objective,

$$J_\pi(\theta) = \sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D},\boldsymbol{a}\sim\pi_{\theta_k}} \left[\min\left(r_k(\theta) A_{\pi_{\theta_k}}(\boldsymbol{o},\boldsymbol{a}), \text{clip}(r_k(\theta), 1\pm\tau) A_{\pi_{\theta_k}}(\boldsymbol{o},\boldsymbol{a}) \right) \right], \tag{2}$$

where $r_k(\theta) = \frac{\pi_\theta(a^i|\boldsymbol{o}^i)}{\pi_{\theta_k}(a^i|\boldsymbol{o}^i)}$, θ_k is the policy parameter at iteration k, and N denotes the number of agents. During training, MAPPO employs a centralized critic network that uses the joint observation to estimate the value function V_w. This centralized value function provides a more stable and informative advantage estimate for updating each agent's policy. The critic parameters are trained to minimize the temporal-difference error using aggregated trajectories from all the agents. The critic is updated separately using the objective

$$J_V(w) = -\sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D}} \left[\left(V_w(\boldsymbol{o}_t) - \hat{R}_t \right)^2 \right], \tag{3}$$

where $\hat{R}_t$ is the discounted reward.

2.3 Reconstruction-based representation learning and Autoencoders

Reconstruction is a self-supervised learning method to learn low-dimensional representations from images. It utilizes an autoencoder, typically consisting of a convolutional encoder g_ϕ that maps an image $\boldsymbol{o}$ to a low-dimensional latent vector $\boldsymbol{z}$ and a deconvolutional decoder f_ψ that reconstructs the image $\boldsymbol{o}$ from the latent vector $\boldsymbol{z}$. Early work used a deterministic autoencoder (Hinton & Salakhutdinov, 2006; Vincent et al., 2008), where the latent vector $\boldsymbol{z}$ is deterministic. Variational autoencoders (VAEs) were introduced by (Kingma & Welling, 2013) and greatly improved representation capacity. In VAEs, the latent vector $\boldsymbol{z}$ is a random variable and we assume a posterior distribution $q_\phi(\boldsymbol{z} \mid \boldsymbol{o})$ over $\boldsymbol{z}$ given the observation $\boldsymbol{o}$ as well as a prior distribution $p(\boldsymbol{z})$ over $\boldsymbol{z}$, which typically is a standard multivariate Gaussian $\mathcal{N}(0, I)$. We can further encourage the disentanglement of the latent representation by setting a large regulariztion in the latent space using a β-VAE (Higgins et al., 2017), whose objective function is

$$\mathcal{L}_{\beta\text{-VAE}}(\psi,\phi) = \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D}}\Big[\mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z}|\boldsymbol{o})}\big[\log p_\psi(\boldsymbol{o} \mid \boldsymbol{z}) \big] \;-\; \beta\, D_{\text{KL}}\big(q_\phi(\boldsymbol{z} \mid \boldsymbol{o} \,\|\, p(\boldsymbol{z})\big) \Big], \tag{4}$$

where $\beta \in \mathbb{R}$ controls the regularization on the latent space, i.e., the ***Kullback-Leibler (KL)*** divergence between the posterior and the prior. A larger β enforces more regularization and tends to disentangle the latent vector more. However, Yarats et al. (2021) show that a smaller β can improve the downstream single-agent RL performance when jointly training the autoencoder with an off-policy RL algorithm.

3 Reconstruction-based representation learning for MARL

Following Yarats et al. (2021), we integrate a β-VAE with MAPPO and jointly optimize the autoencoder and MARL networks, including local actors and a shared critic. We specifically choose MAPPO over other off-policy MARL algorithms, such as MADDPG (Lowe et al., 2017), primarily for practical reasons. Although off-policy algorithms tend to be more sample-efficient by reusing samples from the replay buffer, their training cannot be as easily parallelized as that of on-policy algorithms like MAPPO, which enables the efficient use of computing resources and, in turn, yields better computational efficiency. We show β-VAE's positive effect on improving sample efficiency for vision-based multi-agent control tasks by simply reducing the regularization on the latent space of the β-VAE. Although this simple application of β-VAE already improves the sample efficiency of MAPPO by a large margin, we show in section 3.2 that representations learned from such a method are not consistent across steps. This motivates the development of MA-TRVAE in section 4.

3.1 β-VAE for MARL

We show the performance of MAPPO with β-VAE in two multi-agent quadcopter control tasks, with local RGB observations $o^i \in \mathbb{R}^{64\times64\times4}$ and local PID actions $a^i \in \mathbb{R}$ controlling the vertical motion

of the drone, which is the easiest action setting the environment allows. In these tasks, four agents must cooperate to achieve different goals using their ego-centric view. In the flock-formation task, agents need to track the altitude of the preceding agent. In the leader-follower task, the agents need to stay close to the leader. In both tasks, the leader must fly to a target position. We provide further details and an example of the agent's observation in appendix C.1.

We incorporate a shared convolutional encoder g_ϕ and a deconvolutional decoder f_ψ into MAPPO and optimize them with the objective $\mathcal{L}_{\beta-\text{VAE}}$ given in equation 4 on rollouts collected by the current policy. The actor and centralized critic are trained with latent states $\boldsymbol{z}_t^i$ and $[\boldsymbol{z}_t^1, ..., \boldsymbol{z}_t^N]$ respectively, where $\boldsymbol{z}_t^i \sim g_\phi(\boldsymbol{o}_t^i)$. Using rollouts collected from all agents with the current policy, the shared encoder g_ϕ is updated using gradients from the actor loss as given in equation 2, critic loss as given in equation 3, and the β-VAE loss as given in equation 4, unlike Yarats et al. (2021), who exclude the actor's gradient when updating the encoder. We update the shared encoder after every policy update, along with the decoder, since Yarats et al. (2021) empirically showed a positive correlation between encoder update frequency and RL agent performance. We repeat this joint training of the β-VAE and actor–critic network until convergence. Also, following the observation by Yarats et al. (2021) that a small value of β improves the stability and task performance, we conduct experiments for varying β values and confirm that a less constrained latent space also enhances performance in MARL, as shown in figure 1.

Interestingly, we see a large gap between different β values, which is not observed in single-agent systems (Yarats et al., 2021). Specifically, MAPPO that employs a β-VAE with β values of 1 and 0.1 performs substantially worse than MAPPO without representation learning, whereas MAPPO with smaller β values yields a large performance gain. This suggests that strong regularization on the latent space actually hurts the representation learning in multi-agent systems. We hypothesize that this effect arises from heterogeneous observations across agents. Unlike single-agent systems, where an agent receives a fixed view and global observation of the system, each agent in multi-agent systems perceives the environment only partially from its own ego-centric view. The autoencoder must compress this diverse high-dimensional distribution of observations into a low-dimensional latent space. When the latent space is over-regularized, it may lack the expressive capacity to capture this complexity.

Another trend we observe in figure 1 concerns training stability. Specifically, the variance of episodic rewards decreases as β decreases. This aligns with findings from single-agent systems (Yarats et al., 2021), which reported that the stochasticity of a β-VAE degrades RL agent performance. Indeed, a stochastic representation increases variance in the underlying RL policy. However, when the latent space is more stochastic, namely, when $\beta = \{1, 0.1\}$, the variance in MARL agents' performance is even smaller than with an extremely small β value, e.g., 10^{-7}. This seemingly contradictory observation motivates us to investigate instability in MARL performance further. In the next section, we show that stochasticity in the representation is not the sole cause of instability; weak constraints on the latent space also contribute to this issue.

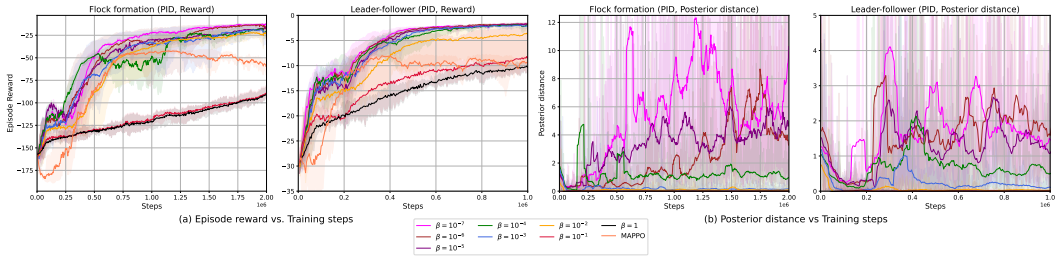


Figure 1: Episode reward and posterior distance vs. training steps using different β values. *(a) shows the episode reward of MAPPO with β-VAE using varying β. (b) measures encoder representation consistency across steps using the KL divergence between posteriors at two consecutive steps, given the same observation, for different β. As β decreases, the performance of underlying MAPPO is improved in (a), but it leads to less consistent representations over steps as shown in (b).*

3.2 INSTABILITY OF REPRESENTATIONS

We investigate another source of instability by understanding how the representation changes over steps. Recall that the encoder and actor-critic are updated alternately for each policy update step. This means that the representation used for policy learning changes between consecutive steps. If the latent space shifts drastically after an encoder update, the representation for the same observation can vary significantly. This undermines the sample efficiency of the underlying MARL agents, since the policy may perceive the same observation inconsistently and must remap them to the same optimal action. To study this effect, we track how the posterior of the latent vector $\boldsymbol{z}$ for a fixed observation evolves over steps, and hypothesize that weaker constraints on the latent space should lead to a larger posterior deviation between successive steps.

In figure 1, we plot the KL divergence between posteriors at two successive steps for the same observation throughout training, which we call posterior distance. When β is large, the posterior distance is close to zero. This is expected, since a large β strongly pushes the posterior towards $\mathcal{N}(0, I)$. By contrast, under weaker constraints, the divergence grows faster, suggesting that the representation of the same observation changes more across updates. Intuitively, this means that the representation of the same observation, instead of being concentrated in a close latent region, is more likely to scatter around the latent space. This investigation explains why MARL agents perform more stably with larger β, as stronger regularization in the latent space mitigates representation drift.

Building on this discovery, we further ask: *can we improve representation consistency when using a small value of β?* Achieving this should further enhance the sample efficiency of the MARL agents, since the policy and critic networks receive more consistent representations over training steps. This motivates trust-region optimization. In the next section, we discuss how to adapt this technique to stabilize representation learning with a small β while preserving the computational efficiency of the β-VAE.

4 TRUST-REGION AUTOENCODER FOR MARL

Trust-region methods are widely used to regularize policy updates in RL. The same idea can be applied directly to representation learning in MARL, where we update the posterior over the latent after each policy update. Formally, the application of a trust region method can be defined as a constrained optimization problem given the objective function in equation 4,

$$\begin{aligned} &\max_{\psi,\phi} \ \mathcal{L}_{\beta-\mathrm{VAE}} \\ &\text{s.t. } \mathbb{E}_t\big[D_{KL}\big(q_\phi(\mathbf{z}_t|\mathbf{o}_t) \,\|\, q_{\phi_{old}}(\mathbf{z}_t|\mathbf{o}_t)\big)\big] \leq \delta, \end{aligned} \tag{5}$$

with constraint $\delta \in \mathbb{R}$, and $q_{\phi_{\text{old}}}$ the posterior inferred by the encoder from the previous step. Although both the objective function and the constraint are based on expected values, which can be approximated by a Monte Carlo estimator using observations collected by agents, solving this constrained optimization problem requires techniques beyond standard backpropagation. For example, TRPO (Schulman et al., 2015) relies on a conjugate gradient method, a second-order optimization approach. This added complexity makes implementation more challenging and the method more computationally demanding, both of which can hinder the application of MA-TRVAE. To avoid this, we instead enforce the trust region constraint by introducing a regularization term into the objective function of the β-VAE, yielding an unconstrained optimization problem with a surrogate

$$\begin{aligned} \mathcal{L}_{\mathrm{TRVAE}} = \mathbb{E}_{\mathbf{o}_t\sim\mathcal{D}}\big[&\mathbb{E}_{\mathbf{z}_t\sim q_\phi(\mathbf{z}_t|\mathbf{o}_t)}\left[\log p_\psi(\mathbf{o}_t \mid \mathbf{z}_t)\right] \\ &- \beta_1 D_{\mathrm{KL}}(q_\phi(\mathbf{z}_t \mid \mathbf{o}_t) \,\|\, p(\mathbf{z}_t)) \\ &-\beta_2 D_{\mathrm{KL}}(q_\phi(\mathbf{z}_t|\mathbf{o}_t) \,\|\, q_{\phi_{\text{old}}}(\mathbf{z}_t|\mathbf{o}_t))\big], \end{aligned} \tag{6}$$

where $\beta_2 \in \mathbb{R}$ controls the strength of the trust-region regularization, and β_1 corresponds to β in the original β-VAE. With this objective function, MA-TRVAE regulates the autoencoder's updates within a trust region while remaining computationally efficient and easy to implement via backpropagation. We summarize MA-TRVAE in figure 2.

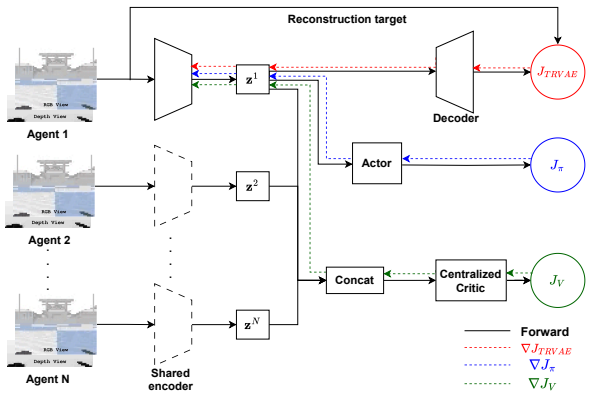


Figure 2: Schematic diagram of the proposed MA-TRVAE architecture. *The dotted encoder boxes represent deep copies of a single shared encoder across agents' local actor and central critic. The parameter update paths are illustrative—in implementation, the encoder is shared and updated with the gradients from the reconstruction J_{TRVAE}, value function J_V, and policy J_π objectives as given in equation 6, equation 3, and equation 2 respectively.*

5 Experiments

In this section, we describe the environment and experiments used to evaluate our proposed method. As we focus on ego-centric, vision-based observations, we need MARL environments that support such observations, which are currently rare. Therefore, we consider multi-agent quadcopter control (MAQC) (Panerati et al., 2021) as the main environment for evaluating MA-TRVAE. Within this setting, we aim to verify three main hypotheses regarding MA-TRVAE: (i) Does the regularization term in equation 6 mitigate representation drift and thereby improve sample efficiency? (ii) How well does the method scale with the number of agents? (iii) Is our method more training efficient compared to prior approaches? Since experiments in a single environment lack diversity, we also test MA-TRVAE on Multi-agent MuJoCo (Peng et al., 2021) and Google research football (Kurach et al., 2020), even though they do not exactly meet our requirements. The former is a MARL environment with state-based observations, and the latter is a visual MARL environment with global observations. Additionally, we recognize the generality of the trust-region VAE and demonstrate its impact in single-agent systems. We refer interested readers to appendix H for results and discussion.

5.1 Experiment setup and Baselines

Vision-based MARL environments. We choose MAQC as the main evaluation environment since it satisfy all assumptions in section.2.1. Each agent in the MAQC environment receives an RGB video frame $\in \mathbb{R}^{64\times48\times4}$ as an observation. A camera mounted on the drone captures the observations, oriented toward the positive direction of the local x-axis. The environment provides three action settings: PID, DYN, and RPM, arranged in order of increasing difficulty. We evaluate MA-TRVAE in two cooperative multi-agent tasks, `flock` and `leader-follower`, across two action settings with 4 agents. More details are provided in appendix C.1.

Additionally, we test MA-TRVAE on two tasks in Google research football, `3vs1-with-keeper` and `pass-shoot-with-keeper`. This environment supports pixel observations. However, it provides a single global observation to all agents, different from the MAQC environment that has an ego-centric view for each agent. See appendix C.3 for more details.

State-based MARL environment. Given the limited availability of vision-based MARL environments, we additionally evaluate our method on a state-based MARL environment Multi-agent MuJoCo, where the observation is raw state information for each agent. See appendix C.2 for more details.

Table 1: Mean and standard deviation of the final episode reward in the vision-based environment. *MA-TRVAE outperforms all MAPPO variants in most tasks. In addition, MA-TRVAE shows the best overall stability, which we attribute to improved representation consistency from trust-region regularization.*

	MAPPO+TRVAE (ours)	MAPPO+MA2CL	MAPPO	MAT
Flock-PID	**-13.73** ± **0.96**	-52.71 ± 10.00	-55.25 ± 2.80	-18.65 ± **0.96**
Flock-DYN	-74.69 ± **3.10**	-121.24 ± 4.82	-89.10 ± 7.55	**-48.69** ± 4.66
Leader-PID	**-1.81** ± **0.09**	-8.24 ± 1.79	-9.62 ± 0.82	-2.05 ± 0.14
Leader-DYN	-14.09 ± 1.49	-20.91 ± 2.93	-16.01 ± **0.45**	**-12.2** ± 2.56
3vs1-with-keeper	**0.63** ± 0.40	0.60 ± **0.27**	0.28 ± 0.96	–
Pass-shoot-with-keeper	0.84 ± **0.03**	**0.96** ± 0.06	0.41 ± 0.91	–

Baselines. We compare MA-TRVAE to different versions of MAPPO (Yu et al., 2022): (i) the standard MAPPO, to demonstrate the performance gains through the VAE with trust-region, and (ii) MAPPO with masked attentive contrastive learning (MA2CL) (Song et al., 2023), which has reported state-of-the-art performance in the MAQC environment. However, because MA2CL performs unstably in our experiments, we also compare against Multi-agent Transformer (MAT) (Wen et al., 2022), a state-of-the-art MARL algorithm that uses a transformer-based policy to learn representations implicitly and often outperforms MAPPO. All methods that take pixel inputs use the same convolutional neural network (CNN) to obtain representations. For each algorithm and task, we independently run five experiments with random seeds to obtain the mean and standard deviation of various evaluation metrics, such as episode rewards and posterior distance. The hyperparameters are kept consistent with those used in the original papers for a fair comparison, and remain the same across all experiments. Detailed hyperparameters are in appendix E.

5.2 Results and Discussion

Stability of representation. We conduct an empirical study to show the effectiveness of the trust-region regularization in stabilizing the representation learning. The empirical study is conducted in `flock` task with PID action setting. We measure the posterior distance by KL divergence between posteriors across two consecutive steps, given the same observation. As shown in figure 3, introducing trust-region regularization reduces posterior distance, and stronger regularization further lowers it. This suggests that the regularization term in equation 6 effectively improves consistency in representations across training steps.

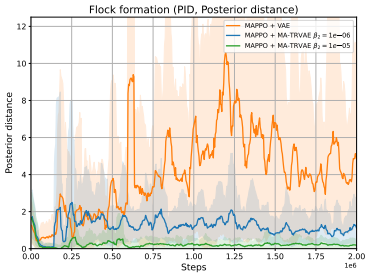


Figure 3: Posterior distance vs. training steps in the MAQC flock task. *Representation is more consistent with heavier trust-region regularization.*

Sample efficiency and stability. To test whether more consistent representations improve sample efficiency and stability in MARL, we evaluate final episode rewards and learning curves in MAQC. As shown in table 1, MA-TRVAE achieves the best final mean reward among all MAPPO-based methods across all MAQC tasks and action settings. It is also competitive with MAT, except in Flock-DYN. MA-TRVAE further achieves the lowest final performance variance in three of four settings, indicating improved stability. Among visual-input MAPPO variants, MA-TRVAE consistently achieves higher sample efficiency and reduces training variance, supporting our hypothesis that more consistent representations improve sample efficiency and stability. We provide additional noisy-observation experiments in Appendix F.4.

We further evaluate generality in the Google Research Football and state-based Multi-agent MuJoCo environments. In Google Research Football, MA-TRVAE slightly underperforms MA2CL on average but substantially improves upon MAPPO, as we see from table 1, showing its effectiveness beyond quadcopter control tasks. We provide learning curves in appendix C.3. In Multi-agent MuJoCo, where all methods use a multi-layer perceptron (MLP) encoder, MA-TRVAE outperforms MAPPO and MAPPO+MA2CL on most tasks as shown in figure 4, suggesting that MA-TRVAE is a general-purpose representation-learning method for MARL.

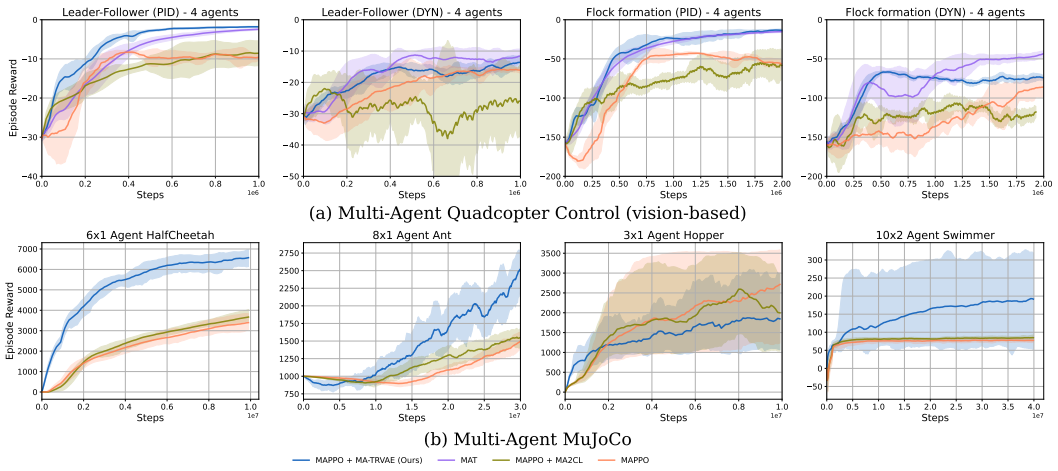


Figure 4: Episode reward vs. training steps for the multi-agent tasks in vision-based MAQC and state-based Multi-agent MuJoCo environments. *MA-TRVAE performs better than other baselines in stability in the MAQC environment and shows competitive sample efficiency to MAT. MA-TRVAE also achieves higher sample efficiency than other baselines in the Multi-agent MuJoCo environment, despite not being designed for state-based observations.*

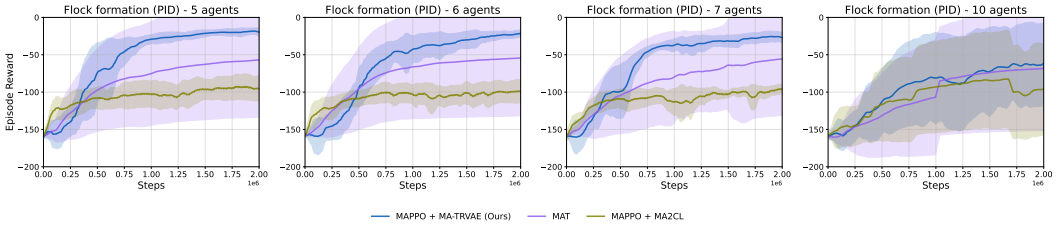


Figure 5: Scaled episode reward vs. training steps for flock formation task, with PID action setting, for increasing number of agents. *MA-TRVAE scales well with the number of agents, showing a stable learning curve, outperforming baselines.*

Scalability. Next, we evaluate MA-TRVAE with 5, 6, 7, and 10 agents in MAQC to assess scalability. As shown in figure 5, MA-TRVAE maintains strong performance while remaining sample-efficient and stable for up to 7 agents. Notably, MA2CL and MAT drop in performance when scaling from 4 to 5 agents. As the number of agents increases further, MA-TRVAE exhibits only a slight decline in performance and remains stable, whereas MAT shows large performance variance. When we scale to 10 agents, performance drops for all methods. These results support the second hypothesis: MA-TRVAE scales well with a small number of agents but requires further exploration to support more agents in MARL.

Training efficiency. Lastly, we conduct experiments to understand how fast MA-TRVAE trains compared to baselines. We run these experiments in both `flock` and `leader-follower` tasks, with the same PID action setting and hardware configuration (see details in appendix E). We plot the episodic return against the running time for each method in figure 6. As MA-TRVAE employs a simple reconstruction method, its training time increases by only 10% compared to MAPPO, thanks to the low computational cost of CNNs. Nevertheless, MA-TRVAE achieves the same performance in less training time than MAPPO, showing better wall-clock efficiency. In contrast, MA2CL and MAT use attention modules that require about 20% more training time than MA-TRVAE. This result verifies our third hypothesis that MA-TRVAE is more training-efficient than the baselines.

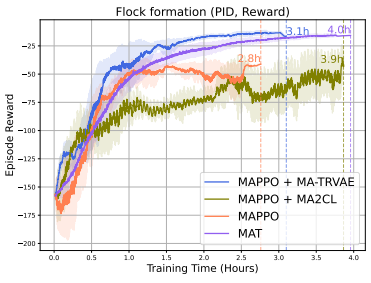


Figure 6: Episode reward vs. training time.

5.3 Ablation Study

Effect of trust-region regularization. To investigate the effect of trust-region regularization, we compare MA-TRVAE to (i) MAPPO with a vanilla VAE (Kingma & Welling, 2022), which is equivalent to MA-TRVAE with $\beta_2 = 0$; (ii) MAPPO with a deterministic autoencoder, which improves sample efficiency in single-agent systems (Yarats et al., 2021). We conduct experiments in all tasks and action settings in the MAQC environment and show the result from `flock` task with DYN action setting in figure 7 (see full task comparison in appendix F.1).

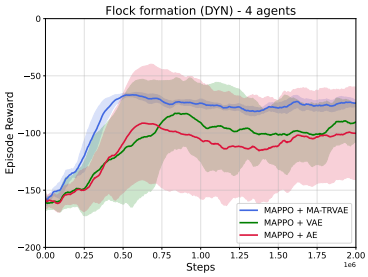


Figure 7: Ablation study of the trust-region regularization.

We observe that all methods show good sample efficiency in mean episode reward, suggesting the effectiveness of reconstruction-based representation learning in vision-based MARL. However, MAPPO + VAE and MAPPO + AE show high performance variance, while MA-TRVAE is the most stable across runs, likely due to improved representation consistency from trust-region regularization. These findings also hold when scaling to more agents; see the result in appendix F.3.

Effect of trust-region regularization strength. Since the core contribution of MA-TRVAE is the trust-region regularization in equation 6, we test the sensitivity of β_2 to understand the effect of trust-region regularization strength. We conduct an experiment on the flock formation task with the PID action setting. We vary β_2 from 1×10^{-2} to 1×10^{-8} and keep the other hyperparameters constant. Fig. 8 shows the performance of MA-TRVAE with different β_2 values. A large β_2 leads to more stable training but poorer performance, whereas a smaller β_2 leads to less stable training but better performance. This is attributed to either under- or over-regularized representations. A large β_2 stabilizes the representation but limits exploration in the latent space, as it is forced to remain close to the previous one. A small β_2, by contrast, allows more exploration in the latent space and tends to perform better for MARL.

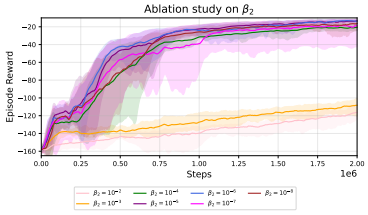


Figure 8: Sensitivity of β_2 values.

Comparison to EMA encoder. While our approach is motivated by the trust-region idea, we note that it is conceptually close to consistency regularization in Mean Teacher (Tarvainen & Valpola, 2017), where a student network with a noisy perturbation is encouraged to produce predictions close to those of an exponential moving average (EMA) teacher network. To relate these two methods, one can view our trust-region approach as a special case of the consistency regularization, where we replace an EMA encoder with a previous encoder $q_{\phi_{\text{old}}}$. Empirically, both approaches mitigate representation drift and improve MARL agent performance. See appendix G for details.

6 Conclusion and Limitations

In this paper, we consider MARL with vision input using reconstruction-based representation learning. We integrate β-VAEs with MAPPO and identify representation drift as a key problem. We further propose MA-TRVAE to mitigate representation drift with trust-region regularization. This yields a simple, training-efficient method. Experiments show that this method is more sample-efficient and stable than MAPPO-based baselines, requiring only 10% more training time than MAPPO. We then scale the method to more agents and demonstrate consistent performance for a small number of agents.

Limitations. The simplicity, training efficiency, and scalability of MA-TRVAE suggest potential applicability to a wide range of multi-agent systems. However, our study is primarily empirical and does not provide a theoretical performance analysis. In addition, because visual MARL environments that follow the Dec-POMDP setting are limited, we evaluate the method on a restricted set of tasks. Developing more comprehensive visual MARL benchmarks, therefore, remains an important direction for future work. Lastly, MARL scalability remains a challenge when scaling to a large number of agents.

AI USE STATEMENT

In this work, we used generative AI tools to implement a TRVAE with EMA encoders (see the experiment in appendix G). We have not used generative AI tools to propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat datasets, support qualitative and thematic data analysis, interpret results and generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, are not applicable to this work. Additionally, we used generative AI tools to search for information and edit the research paper to improve readability. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by 1 author. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

(This section is **recommended** and does not count toward the page limit.)

If authors feel that their paper submission raises questions regarding the Code of Ethics, they are encouraged to include a paragraph of Ethics Statement (at the end of the main text before references) to address potential concerns where appropriate. Topics include, but are not limited to, studies that involve human subjects, practices to data set releases, potentially harmful insights, methodologies and applications, potential conflicts of interest and sponsorship, discrimination/bias/fairness concerns, privacy and security issues, legal compliance, and research integrity issues (e.g., IRB, documentation, research ethics). This statement should not be more than 1 page.

REPRODUCIBILITY STATEMENT

(This section is **recommended** and does not count toward the page limit.)

It is important that the work published in ICLR is reproducible. Authors are strongly encouraged to include a paragraph-long Reproducibility Statement at the end of the main text (before references) to discuss the efforts that have been made to ensure reproducibility. This paragraph should not itself describe details needed for reproducing the results, but rather reference the parts of the main paper, appendix, and supplemental materials that will help with reproducibility. For example, for novel models or algorithms, a link to an anonymous downloadable source code can be submitted as supplementary materials; for theoretical results, clear explanations of any assumptions and a complete proof of the claims can be included in the appendix; for any datasets used in the experiments, a complete description of the data processing steps can be provided in the supplementary materials. Each of the above are examples of things that can be referenced in the reproducibility statement.

AUTHOR CONTRIBUTIONS

If you'd like to, you may include a section for author contributions as is done in many journals. This is optional and at the discretion of the authors.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

REFERENCES

Joris Dinneweth, Abderrahmane Boubezoul, René Mandiau, and Stéphane Espié. Multi-agent reinforcement learning for autonomous vehicles: A survey. *Autonomous Intelligent Systems*, 2(1): 27, 2022.

Mingxiao Feng, Yaodong Yang, Wengang Zhou, and Houqiang Li. Timar: Transition-informed representation for sample-efficient multi-agent reinforcement learning. *Neural Networks*, 184: 107081, 2025. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2024.107081.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, NIPS'18, pp. 2455–2467, Red Hook, NY, USA, 2018a.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018b.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2019a.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels, June 2019b. URL http://arxiv.org/abs/1811.04551. arXiv:1811.04551 [cs].

Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, pp. 1–7, 2025.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Irina Higgins, Arka Pal, Andrei A. Rusu, Loic Matthey, Christopher P. Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. DARLA: Improving Zero-Shot Transfer in Reinforcement Learning, June 2018. URL http://arxiv.org/abs/1707.08475. arXiv:1707.08475 [stat].

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Yingdong Hu, Renhao Wang, Li Erran Li, and Yang Gao. For Pre-Trained Vision Models in Motor Control, Not All Policy Learning Methods are Created Equal, 2023. URL https://arxiv.org/abs/2304.04591v2.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, pp. 267–274, 2002.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022. URL https://arxiv.org/abs/1312.6114.

Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2021.

Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. Google research football: A novel reinforcement learning environment. In *AAAI Conference on Artificial Intelligence*, pp. 4501–4510, 2020.

Sascha Lange and Martin Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *International Joint Conference on Neural Networks*, pp. 1–8, July 2010. doi: 10.1109/IJCNN.2010.5596468. URL https://ieeexplore.ieee.org/document/5596468. ISSN: 2161-4407.

Sascha Lange, Martin Riedmiller, and Arne Voigtländer. Autonomous reinforcement learning on raw visual input data in a real world application. In *International Joint Conference on Neural Networks*, pp. 1–8, June 2012. doi: 10.1109/IJCNN.2012.6252823. URL https://ieeexplore.ieee.org/document/6252823. ISSN: 2161-4407.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pp. 5639–5650, 2020.

Alex X. Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model, October 2020. URL http://arxiv.org/abs/1907.00953. arXiv:1907.00953 [cs].

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, 2017.

Chengdong Ma, Aming Li, Yali Du, Hao Dong, and Yaodong Yang. Efficient and scalable reinforcement learning for large-scale network control. *Nature Machine Intelligence*, 6(9):1006–1020, 2024.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards Universal Visual Reward and Representation via Value-Implicit Pre-Training, 2023. URL http://arxiv.org/abs/2210.00030. Comment: ICLR 2023, Notable-Top-25% (Spotlight). Project website: https://sites.google.com/view/vip-rl.

Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals, December 2018. URL http://arxiv.org/abs/1807.04742. arXiv:1807.04742 [cs].

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. 2022.

Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. Springer Publishing Company, Incorporated, 1st edition, 2016. ISBN 3319289276.

James Orr and Ayan Dutta. Multi-agent deep reinforcement learning for multi-robot applications: A survey. *Sensors*, 23(7):3625, 2023.

Jacopo Panerati, Hehui Zheng, SiQi Zhou, James Xu, Amanda Prorok, and Angela P. Schoellig. Learning to fly—a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 7512–7519, 2021. doi: 10.1109/IROS51168.2021.9635857.

Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The Unsurprising Effectiveness of Pre-Trained Vision Models for Control, 2022. URL http://arxiv.org/abs/2203.03580. Comment: First two authors contributed equally.

Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. Facmac: Factored multi-agent centralised policy gradients. *Advances in neural information processing systems*, 34:12208–12221, 2021.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4295–4304, 2018.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017a.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017b.

Wenling Shang, Lasse Espeholt, Anton Raichuk, and Tim Salimans. Agent-centric representations for multi-agent reinforcement learning, 2021.

Haolin Song, Mingxiao Feng, Wengang Zhou, and Houqiang Li. Ma2cl:masked attentive contrastive learning for multi-agent reinforcement learning. In *International Joint Conference on Artificial Intelligence*, pp. 4226–4234, 2023.

Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *International conference on machine learning*, pp. 9870–9879. PMLR, 2021.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. volume 30, 2017.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine learning*, pp. 1096–1103, 2008.

Muning Wen, Jakub Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. In *Advances in Neural Information Processing Systems*, pp. 16509–16521, 2022.

Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control, 2022. URL https://arxiv.org/abs/2203.06173.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *AAAI Conference on Artificial Intelligence*, pp. 10674–10681, 2021.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Zhecheng Yuan, Zhengrong Xue, Bo Yuan, Xueqian Wang, Yi Wu, Yang Gao, and Huazhe Xu. Pre-Trained Image Encoder for Generalizable Visual Reinforcement Learning. 2022. doi: 10.48550/ARXIV.2212.08860.

A EXTENDED RELATED WORK

MARL. MARL extends reinforcement learning to environments with multiple interacting agents, where objectives can be cooperative, competitive, or a combination of both. A straightforward strategy is to train agents independently, but such methods suffer from non-stationarity because each agent's policy updates alter the effective environment dynamics. To mitigate this issue, *centralized training with decentralized execution* (CTDE) has become the dominant paradigm. In CTDE, a centralized critic utilizes global state information during training, whereas decentralized actors rely solely on local observations during execution. Lowe et al. (2017) introduced this framework for continuous control by extending deterministic policy gradients with a centralized critic, known as the *multi-agent deep deterministic policy gradient* (MADDPG). This paradigm has since become the foundation for subsequent methods. Policy-gradient-based extensions of CTDE have proven particularly effective. Yu et al. (2022) adapted PPO to the multi-agent setting with a centralized value function, achieving stable and scalable performance in cooperative settings, known as multi-agent proximal policy optimization (MAPPO). Building on this, Kuba et al. (2021) extends trust-region methods to heterogeneous agents called heterogeneous-agent trust-region policy optimization (HATPRO) and heterogeneous-agent proximal policy optimization (HAPPO), which provides theoretical guarantees of monotonic policy improvement under CTDE and enhances sample efficiency. Value decomposition offers an alternative approach to coordination and credit assignment by factorizing a global action-value function into per-agent utilities. Rashid et al. (2018) introduce such a method called QMIX, which factorizes a global action-value function into per-agent utilities via a monotonic mixing network, enabling effective coordination in cooperative settings while maintaining scalability. More recently, transformer-based architectures have been proposed to capture temporal and inter-agent dependencies better. Wen et al. (2022) propose multi-agent transformer (MAT) that treats agents and their trajectories as sequences, applying self-attention to model interactions and dynamics. This approach offers expressive joint representations and has demonstrated enhanced performance, though at the expense of increased computational complexity. Together, these methods represent the major algorithmic families that have shaped MARL in recent years. While they have advanced the state of the art, challenges remain in scalability, sampspace using an attentive model and contrastive502 loss. This approachle efficiency, and robustness in high-dimensional and partially observable environments, motivating the exploration of complementary directions such as representation learning.

Representation learning in RL and MARL. Early approaches to representation learning for model-free RL in the single-agent case apply deep autoencoders to learn feature spaces (Lange & Riedmiller, 2010; Lange et al., 2012), but lack scalability in complex environments and require expert knowledge. Subsequent methods leverage VAEs (Kingma & Welling, 2022) for this task, where the RL agent learns policies using the latent representation as inputs (Higgins et al., 2018; Nair et al., 2018). Reference (Yarats et al., 2021) then identify that the stochasticity of VAEs damages the RL agent performance and propose to use a deterministic autoencoder. Their method, dubbed SAC+AE, was reported as achieving state-of-the-art performance at the time. Apart from reconstruction-based approaches, (Laskin et al., 2020) and (Stooke et al., 2021) leverage contrastive learning, but rely on designing a similarity function as well as positive and negative samples. Model-based approaches leveraging the VAE are based on world models (Ha & Schmidhuber, 2018a), which proved too complex due to several auxiliary losses (Hafner et al., 2019b)(Lee et al., 2020). Apart from end-to-end methods, pretrained methods like like R3M Nair et al. (2022), PVR Parisi et al. (2022), MVP Xiao et al. (2022), VIP (Ma et al., 2023), PIE-G (Yuan et al., 2022) have found success in control and robotics tasks. The key difference to our setting is that these encoders are frozen, and therefore do not experience representation drift as a matter of design, the very problem our trust-region method addresses. Moreover, they are evaluated predominantly robotic manipulation via imitation learning , meaning that their conclusions do not necessarily carry over to the RL, let alone MARL setting (Hu et al., 2023). In the multi-agent domain, Shang et al. (2021) combine a cross-agent attention module with an unsupervised trajectory prediction task. Reference (Song et al., 2023) introduce a masked attentive contrastive learning framework that reconstructs masked agent observations in the latent space using an attentive model and contrastive loss. This approach enables agents to leverage inter-agent correlations, leading to improvements in existing MARL algorithms. Building on this, (Feng et al., 2025) use a attention-based world-model approach with a self-supervised learning objective. While these approaches show promise, they either lack generality, impose high computational demands, or introduce considerable complexity. This motivates the need for simpler methods, that are both computationally efficient and scalable, a gap that we address in this work.

Trust region optimization in RL. Trust region methods are a common tool used in RL to bound the size of policy updates. Kakade & Langford (2002) first introduce this idea into RL by forming a constrained optimization problem and prove a monotonic improvement guarantee for policy update. While their method is based on mixing policies, Schulman et al. (2015) introduce *trust-region policy optimization* (TRPO) that extend their method to non-linear stochastic policies, which enables the use of neural networks for solving high-dimensional control tasks. TRPO enforces the trust region constraint by constraining the *Kullback-Leibler (KL) divergence* between the old policy and the new one. Despite its theoretical rigor, TRPO uses second-order optimization to enforce the constraint and hence scales poorly. Schulman et al. (2017a) later introduce *proximal policy optimization* (PPO) to overcome this issue by forming the trust region optimization as an unconstrained optimization problem. This allows for efficient first-order optimizations and better sample efficiency.

B TRUST REGION OPTIMIZATION

Trust region methods optimize an objective function by restricting the step size of updates so that the new solution does not deviate drastically from the previous one. This is typically achieved by enforcing a constraint in the optimization process or adding a penalty term to the objective function. When it comes to policy optimization, while TRPO implements the idea by solving a constrained optimization problem, which proved to be computationally costly, PPO chooses the latter approach by adding a clip penalty to the objective function as a surrogate to optimize

$$L^{clip}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1 \pm \tau)\, \hat{A}_t \right) \right], \tag{7}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|o_t)}{\pi_{\theta_{old}}(a_t|o_t)}$ is the ratio between the new policy $\pi_\theta(a_t|o_t)$ and the old one $\pi_{\theta_{\text{old}}}(a_t|o_t)$, and $\hat{A}_t$ is the advantage function that estimating how much the current policy is better than average. The clip operator trims the input value to keep it within the interval $[1-\tau, 1+\tau]$, where τ is the clip parameter. Alternatively, one can add a KL penalty as another surrogate objective

$$L^{\text{KL}}(\theta) \ = \ \mathbb{E}_t \Big[r_t(\theta)\, \hat{A}_t \ - \ \beta\, D_{KL}\big(\pi_{\theta_{old}}(\cdot|o_t) \,\|\, \pi_\theta(\cdot|o_t)\big) \Big], \tag{8}$$

where $D_{KL}\big(\pi_{\theta_{old}}(\cdot|o_t) \,\|\, \pi_\theta(\cdot|o_t)\big)$ is the KL divergence between the old policy and the new one, and β is a coefficient controlling the range of the policy update.

C ADDITIONAL DETAILS OF THE MULTI-AGENT ENVIRONMENTS

We elaborate on the details of other multi-agent environments, including MAQC, Multi-agent MuJoCo and Google Research Football, in this section. In addition to it, we briefly explain the tasks we performed.

C.1 ADDITIONAL DETAILS OF THE MAQC TASKS

In this section, we briefly describe the two multi-agent tasks, named flock and leader-follower, in MAQC. Let $\mathbf{x}_i = (x_i, y_i, z_i)$ be the position coordinates, r_i be the individual reward of i-th agent, and $R = \sum_{i=1}^{N} r_i$ be the team reward.

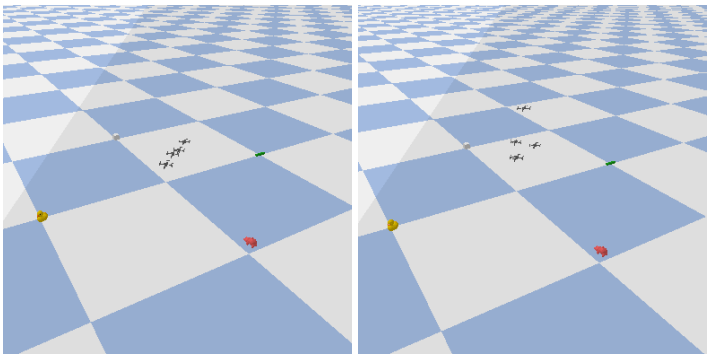
- **Flock**: In the flock scenario of MAQC, the objective is for the first agent to keep its position as close as possible to a predefined location (e.g., p). The individual reward for the first agent is

  $$r_1 = -||\boldsymbol{p} - \mathbf{x}_1||_2^2. \tag{9}$$

  The individual rewards for the remaining agents are determined by their ability to track the altitude of the preceding agent. The reward is defined as

  $$r_i = -(z_i - z_{i-1})^2, \tag{10}$$

  for $i = 2, 3, ..., N$. That is, all drones need to follow the first drone in a line. Figure 9a shows the flock formation task captured during the MARL training.



(a) Flock formation

(b) Leader-follower



(c) Example of the RGB view of an agent.

Figure 9: Multi-agent quadcopter control tasks. (a) shows a training step in Flock-formation task and (b) shows a training step of the Leader-follower task. (c) shows the RGB observation received by one agent.

- **Leader-follower**: In the leader-follower scenario of MAQC, the goal is to train the follower drones to track the leader drone. The leader drone is expected to keep its position as close as possible to a predefined location. The individual reward for the leader drone is
$$r_1 = -||\boldsymbol{p} - \mathbf{x}_1||_2^2. \tag{11}$$
The individual rewards for the follower drones are determined by their ability to track the position of the leader drone. The reward is defined as
$$r_i = -\frac{1}{N}||\mathbf{x}_i - \mathbf{x}_1||_2^2, \tag{12}$$
for $i = 2, 3, ..., N$. That is, all drones need to keep close to the leader drone. figure 9b shows the leader-follower task captured during the MARL training.

There are three action setting in the environment, each controls the drone in different levels. We give details below

- PID: this action specifies a small change in target altitude, and a built-in controller converts it into motor commands while stabilizing the drone;
- DYN: this action specifies total thrust and the environment converts that command into motor speeds;
- RPM: this action directly control the speed for motors.

The PID setting is generally easier because the policy can rely on feedback control. The other two settings require the policy to regulate altitude through lower-level commands and provide no direct attitude correction.

C.2 Multi-agent MuJoCo

Multi-agent MuJoCo is a multi-agent continuous-control benchmark built on top of MuJoCo environments. In this benchmark, a single system is decomposed into multiple agents, each controlling

a subset of joints or actuators. Although the agents act locally, successful task completion requires coordinated behavior across the full system. Therefore, Multi-agent MuJoCo provides a setting for evaluating cooperative multi-agent reinforcement learning methods in continuous observation and action spaces.

In this work, we evaluate MA-TRVAE on four Multi-agent MuJoCo tasks, `6x1 HalfCheetah`, `8x1 Ant`, `3x1 Hopper`, and `10x2 Swimmer`. The notation `NxM` denotes a decomposition in which the robot is divided into N agents, with each agent controlling M actuator dimensions. These tasks cover different locomotion structures and coordination requirements, allowing us to test whether MA-TRVAE can learn useful latent representations across diverse multi-agent continuous-control settings. More details of the task are given below.

- **6x1 HalfCheetah:** In this task, the HalfCheetah is decomposed into six agents, with each agent controlling one actuator. The objective is to coordinate the movement of different body joints so that the robot can achieve stable and efficient forward planar locomotion.
- **8x1 Ant:** In this task, the Ant is decomposed into eight agents, with each agent controlling one actuator. Since the Ant has a quadruped morphology, the agents must coordinate leg movements to maintain body stability while generating forward motion.
- **3x1 Hopper:** In this task, the Hopper is decomposed into three agents, where each agent controls one actuator. The goal is to coordinate the body and leg joints to maintain balance and achieve a forward hopping motion.
- **10x2 Swimmer:** In this task, the Swimmer is decomposed into ten agents, with each agent controlling two actuator dimensions. The objective is to generate coordinated oscillatory body movements across multiple segments so that the agent can produce effective swimming motion.

The reward in Multi-agent MuJoCo is typically based on task progress, such as forward velocity or distance travelled, together with additional terms related to control cost, stability, survival, or energy efficiency depending on the environment. In the multi-agent setting, agents commonly share the same global reward, which encourages cooperative behavior toward the overall locomotion objective.

C.3 Google Research Football

Google Research Football is a football-based reinforcement learning benchmark for evaluating decision-making and coordination in multi-agent environments. In this work, we use the `academy_3_vs_1_with_keeper` task, where three attacking agents cooperate against one defender and one goalkeeper to create a scoring opportunity and score a goal. The `academy_3_vs_1_with_keeper` scenario requires the attacking agents to coordinate their movements, support the ball carrier, avoid the defender, and choose effective actions near the goal. The RGB observation of this setting is shown in figure 10a We also consider the `academy_pass_and_shoot_with_keeper` setting, which focuses on cooperative attacking behavior, where one player must create or use a passing opportunity and another player must receive the ball and shoot against the goalkeeper. The task requires coordinated passing, positioning, and finishing near the goal. A sample RGB observation of this setting is shown in figure 10b



(a) Academy 3 vs 1 with keeper (b) Academy pass and shoot with keeper

Figure 10: Google research football tasks

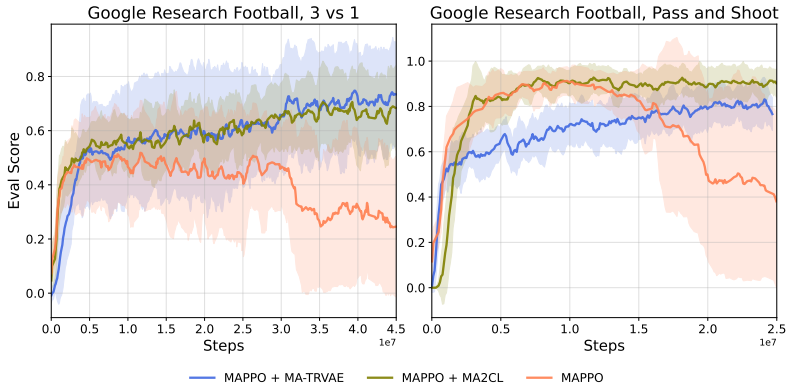


Figure 11: Evaluation score vs. training steps for the academy setting in google research football. *We can observe that MA-TRVAE is performing slightly better when compared to MA2CL and outperforming the unstable MAPPO baseline, which is also the underlying RL algorithm of our method.*

Google Research Football supports different observation representations, including raw RGB pixel observations, grayscale pixel observations, super mini-map (SMM) observations, and vector-based observations. Pixel observations provide visual information from the rendered game screen, SMM observations provide a compact spatial representation of the game state, and vector observations provide structured numerical features such as player positions, ball information, possession, and game-state variables. In the experiment, we use grayscale pixel observations for all methods, as we did not observe any signal of learning for all methods when using raw RGB pixel observations. The learning curve is shown in figure 11.

D Details of MA-TRVAE

In this section, we include a detailed algorithm of our proposed method (i.e., MA-TRVAE combined with MAPPO). The end-to-end training process of the framework is outlined in Algorithm 1, which is heavily based on MAPPO. Also, our code is mainly based on the MAPPO implementation.[2]

E Hyperparameters

Table 2 contains the common hyperparameters used for this paper. Table 3 shows the hyperparameters used for each algorithm in the experiment. For training the MARL agents and conducting other experiments, we utilized a cluster equipped with an A100 GPU and 80 GB of memory. The running time for a single experiment varies depending on the environment and tasks. To produce the main result in the MAQC environment, the running time for different algorithms is between 15 hours and 20 hours. We provide a computing usage form in the supplementary material.

Table 2: Common hyperparameters used for all methods.

Hyper-parameters	Value	Hyper-parameters	Value
γ	0.99	optim eps	1e-6
max grad norm	0.5	gain	0.01
hidden layer dim	64	entropy coef	0.01
use huber loss	True	rollout threads	20
episode length	200	batch size	4000
stacked frames	1	training threads	16

[2]https://github.com/marlbenchmark/on-policy

Algorithm 1 MA-TRVAE for MAPPO

Initialize encoder ϕ, decoder ψ, dummy encoder ϕ_{old}, actor θ, critic w, data buffer $\mathcal{D}$
for each episode **do**
  for each timestep $t = 1$ to T **do**
    for each agent $i = 1$ to N **do**
      Observe o_t^i, encode $z_t^i = g_\phi(o_t^i)$
      Sample action $a_t^i \sim \pi_\theta^i(a|z_t^i)$
    end for
    Execute joint action a_t, store transition in $\mathcal{D}$
  end for
  for each mini-batch from $\mathcal{D}$ **do**
    for each agent i **do**
      Encode latent (μ^i, σ^i), reparametrize z_t^i
      Reconstruct $\hat{o}_t^i$, compute $\mathcal{L}_{\text{TR-VAE}}, \mathcal{L}_{\text{actor}}^i$
    end for
    Compute critic loss $\mathcal{L}_{\text{critic}}$ using global latent state $[z_t^1, ..., z_t^N]$
    Update θ, w with PPO
    Update ϕ, ψ with $\mathcal{L}_{\text{TR-VAE}}$
  end for
  Update $\phi_{\text{old}} \leftarrow \phi$
end for

Table 3: Hyperparameters used for MA-TRVAE, VAE, AE, MA2CL, and MAPPO.

Parameters \ Algorithm	MA-TRVAE	MAPPO + VAE	MAPPO + AE	MA2CL	MAPPO
critic lr	5e-3	5e-3	5e-3	5e-3	5e-3
actor lr	5e-4	5e-4	5e-4	5e-4	5e-4
ppo epochs	5	5	5	5	5
ppo clip	0.2	0.2	0.2	0.2	0.2
num mini-batch	4	4	4	4	4
num hidden layer	2	2	2	2	2
λ	/	/	1e-06	/	/
β_1	1e-07	1e-07	/	/	/
β_2	1e-06	/	/	/	/

F ADDITIONAL RESULTS

This section contains the results of additional experiments we conducted for this paper.

F.1 COMPLETE MAQC RESULTS

The complete result of the experiments performed in the MAQC environment is shown in figure 13, where MA-TRVAE is compared against all the baselines, including other reconstruction-based methods.

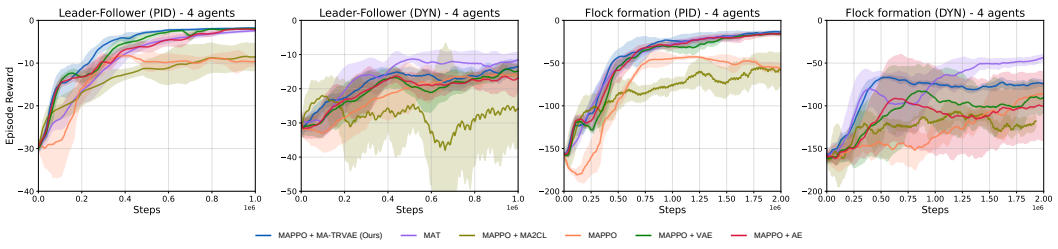


Figure 12: Episode reward vs. training steps for the multi-agent tasks in vision-based MAQC.

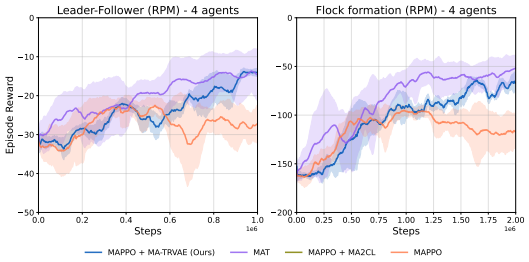


Figure 13: Episode reward vs. training steps for the multi-agent tasks in vision-based with RPM action setting.

F.2 Computational efficiency

The results of the computational efficiency test, performed in both the flock formation and leader-follower tasks with the PID action setting, are shown in figure 14. We can see similar results as we saw in section 5.2.

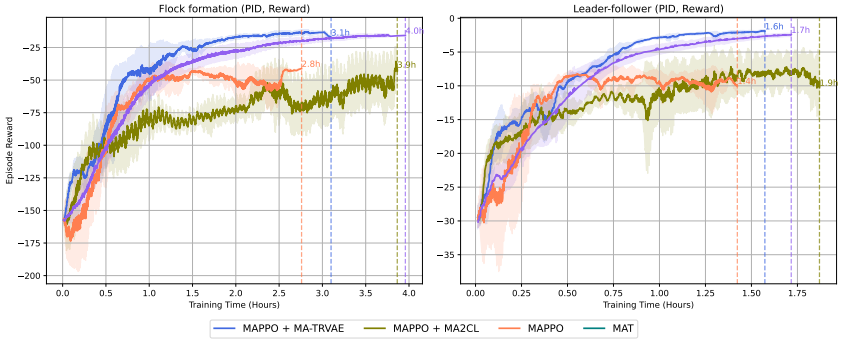


Figure 14: Episode reward vs. training time for flock formation tasks, with PID action setting. *We can see that MA-TRVAE trains faster than the MA2CL, with a better performance. The black dashed line represents the final episodic reward achieved by the MAPPO algorithm trained with proprioceptive state, which is also outperformed by MA-TRVAE.*

F.3 Scalability experiment

The results of the scalability test performed in the leader-follower task with PID action setting are shown in figure 15. Similar to the results obtained from the flock formation experiment as shown in section 5.2, we can see that MA-TRVAE scales better when compared to other baselines.

F.4 Experiment with noisy observations

In the real world, sensors mounted on physical agents are often subject to various noise sources, such as illumination changes, motion blur, occlusion, or sensor artifacts. Therefore, a robust multi-agent learning framework must not only perform well under clear observations but also retain performance when faced with degraded visual input. This experiment investigates the empirical robustness of MA-TRVAE framework, when agents are trained under additive Gaussian noisy observation conditions. This experiment is designed in two different ways, as follows, based on how the noise is added.

- **Shared noise:** A single random Gaussian noise distribution is sampled for each timestep and added uniformly to all agents' observations,

$$\tilde{\boldsymbol{o}}_t^i = \boldsymbol{o}_t^i + \boldsymbol{\xi}, \quad \boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}). \tag{13}$$

  This simulates environment-level noise, such as lighting shifts or global occlusions, that affect all agents identically. The experiment is performed by taking $\sigma = 10$. For the reference, the original pixel observation range is $[0, 255]$.

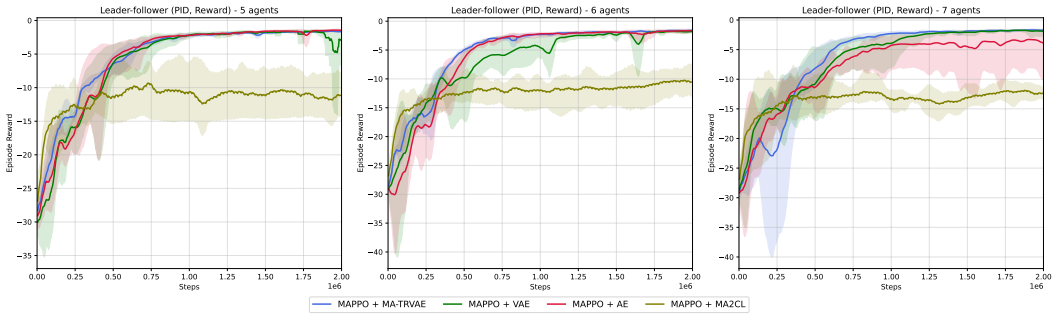


Figure 15: Episodic reward vs. training step plot for the leader-follower task with PID action setting, where the number of agents has been varied from five to seven. *We can observe that MA-TRVAE scales well with the number of agents, with a stable learning curve, outperforming the baselines*

- **Agent-specific noise:** Each agent receives the random Gaussian noise independently with different standard deviations,

$$\tilde{\boldsymbol{o}}_t^i = \boldsymbol{o}_t^i + \boldsymbol{\xi}_i, \quad \boldsymbol{\xi}_i \sim \mathcal{N}(\mathbf{0}, \sigma_i^2 \boldsymbol{I}). \tag{14}$$

  This setting simulates localized sensor noise, which will be different for the agents. The experiment is performed by taking $\sigma_1 = 10$, $\sigma_2 = 5$, $\sigma_3 = 2$, $\sigma_4 = 5$.

These noisy observations $\tilde{\boldsymbol{o}}_t^i$ are passed to the shared encoder g_ϕ, which learns latent representations $\boldsymbol{z}_t^i = g_\phi(\tilde{\boldsymbol{o}}_t^i)$. The rest of the training pipeline remains the same. This experiment is conducted in the flock formation task with the PID action setting as mentioned in section 5.1, over two million training steps. The number of agents is set to four. In figure 16, it can be observed that, MA-TRVAE outperforms other reconstruction-based methods, showing a better empirical robustness under additive Gaussian noise.

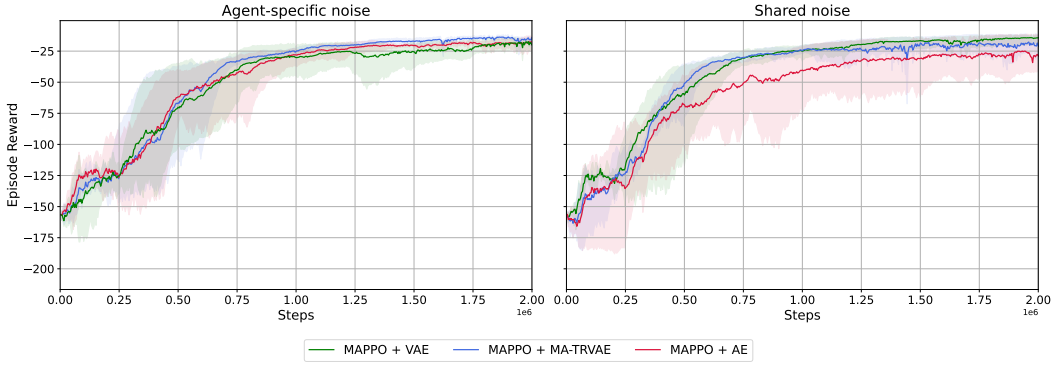
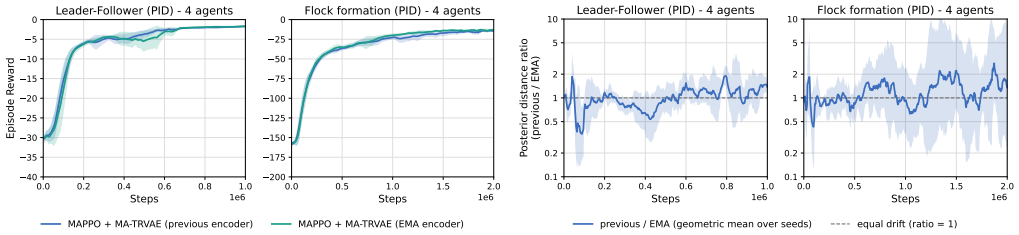


Figure 16: Episodic reward vs. training step plot for the flock formation task with PID action setting, where noise is added to the observations. *We can observe that MA-TRVAE scales well with the number of agents, with a stable learning curve, outperforming the baselines*

G Experiment with EMA encoder

We conducted an experiment of using an EMA target encoder (decay = 0.99) for the trust-region penalty in the Flock formation and Leader-fllower task and compare with our approach that using previous encoder as the target encoder. We use the same β_2 for both approaches. We found this approach is effective in reducing representation drift (see Fig. 17b) and improving the sample efficiency of MAPPO-TRVAE (see Fig. 17a). The performance is similar to that of the previous encoder, but with slightly more variance. We think this behavior can be explained by EMA's average parameter updating, which results in a target encoder that preserves more information from historical



(a) Comparision to EMA encoder

(b) Posterior distance ratio.

Figure 17: Different target encoders mitigate representation drift closely.

changes instead of an encoder from just the previous step. As a result, the "trust-region" penalty serves as a target encoder consistency penalty, producing a representation that is consistent with the history-average encoder rather than the previous encoder. We think this is an effective approach to obtain a more consistent representation for a downstream RL agent. This shows that mitigating the representation drift is an effective way to improve the downstream MARL agent sample efficiency.

H Performance of Trust-region Autoencoder in Single-agent RL

While MA-TRVAE has most potential in the multi-agent domain, it is similarly applicable in single-agent environments and provides similar benefits. To validate that proposition, we provide results from single-agent experiments. It is noteworthy that the underlying RL algorithm is different in the single-agent case than in the multi-agent case. In the latter, the algorithm is based on PPO, an on-policy algorithm, whereas in the former, the RL algorithm is soft actor-critic (SAC), an off-policy algorithm. The reason for this is that we want to evaluate the performance of TRVAE in the single-agent environment against a well-established and fair baseline (Yarats et al., 2021), which uses SAC as the base RL algorithm. Additioanlly, this can show the generality of TRVAE as it can be applied to both on-policy and off-policy RL algorithms.

H.1 Experiments and setup

We evaluate the performance of our method in the single-agent domain (named SAC+TRVAE) against the performance of SAC+AE (Yarats et al., 2021) and a vanilla VAE. For the purposes of the single-agent evaluation, we consider eight different continuous control tasks from the DeepMind control suite (Tassa et al., 2018), specifically `finger_spin`, `reacher_easy`, `ball_in_cup_catch`, `cartpole_swingup`, `walker_walk`, `walker_run`, `cheetah_run`, and `hopper_stand`. We run the algorithms with the same 5 random seeds for each environment, with $\beta_1 = 10^{-7}$ for all experiments.

The setup of these experiments follows the setup and implementation of (Yarats et al., 2021), available on their GitHub page. Their repository did not include an implementation of a VAE, so we modified their code to include an implementation of a VAE and the trust-region optimization.

H.2 Results

H.2.1 Stability and sample efficiency

The results in figure 18 illustrate both the improved performance of the RL agent with the TRVAE over both the vanilla VAE as well as SAC+AE. In `ball_in_cup_catch`, the gains in episodic returns are negligible. However, the variance is improved with the trust-region method, though a single run with $\beta_2 = 10^{-4}$ had some instability. In `cartpole_swingup`, both the vanilla VAE and SAC+TRVAE outperform SAC+AE (a result inconsistent with (Yarats et al., 2021)). Gains in performance are negligible (as with `ball_in_cup_catch`), but the variance of SAC+TRVAE is better. In `cheetah_run`, all methods show similar performance, where SAC+TRVAE slightly outperforms SAC+AE and the vanilla VAE. The variance is again lowest with the trust-region. SAC+TRVAE shows the most impressive gains in `finger_spin` in terms of episodic returns and

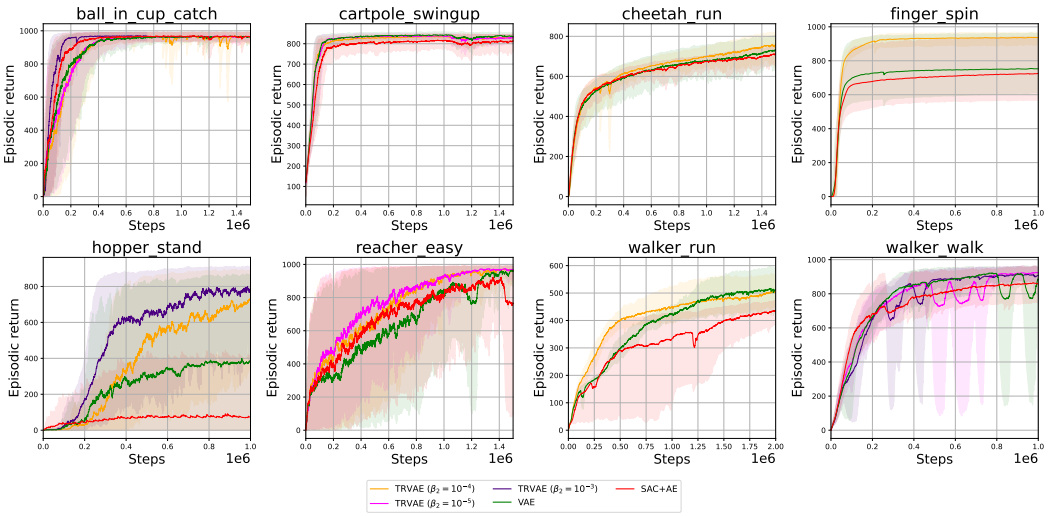


Figure 18: Episodic returns for the SAC+TRVAE compared against SAC+AE (Yarats et al., 2021) a vanilla VAE. The figure shows that the trust-region method achieves similar or better performance across all environments. Furthermore, TRVAE improves the variance of the runs.

variance when compared with a vanilla VAE and SAC+AE. In one environment (`hopper_stand`), SAC+AE failed to learn effectively, whereas SAC+TRVAE shows impressive performance, as well as lower variance, compared to its vanilla counterpart. In `reacher_easy`, SAC+TRVAE outperforms other methods with several hyperparameter settings. This environment does exhibit high variance in all methods; however, the trust-region method is the most stable. In `walker_walk`, all methods showed some instability, where singular runs had bizarre dips in performance. However, on average, SAC+TRVAE has the least variance with gains in performance over SAC+AE. Interestingly, the VAE also beats SAC+AE (once again inconsistent with the original paper). In `walker_run` SAC+TRVAE has the best sample efficiency, but across $2 \cdot 10^6$ steps shows similar performance to the vanilla VAE (though with less variance), once more beating SAC+AE. In summary, our experiments demonstrate an increase in sample efficiency across all environments in which the algorithms were evaluated. SAC+TRVAE method exhibits the lowest variance (i.e., better stability) and improved performance in some. Interestingly, some of our findings on the performance of SAC+AE are inconsistent with those presented in the original paper.

H.2.2 Consistency of Representations

The posterior loss can be loosely utilized as a proxy for the representation drift, i.e., the higher the loss, the more the representations change between iterations. The posterior loss is the trust-region term of the TRVAE, which is to say we measure the consistency of representation by the KL divergence between the posterior distributions of successive encoders when encoding the *current observations*. This measurement is done by maintaining a frozen copy of the encoder parameters from the previous step, encoding the current observations using both sets of parameters, and then measuring the KL divergence of the resulting distributions. figure 19 illustrates the posterior losses for the environments. We observe that the posterior loss of the vanilla β-VAE is the highest, as there is nothing to explicitly constrain the posterior distributions between update steps. Further, we see that the higher the β_2 value (i.e., the stronger the regularization on the latent update), the smaller the posterior loss. In `ball_in_cup_catch` we observe a spike in the posterior loss with $\beta_2 = 10^{-4}$. However, across steps, the posterior loss is smaller. The explanation for the spike is unknown, but it may be due to numerical instability or a random coincidence, as we also observe such spikes with the VAE.

It is noteworthy that a minimal posterior loss is not necessarily desirable. Too strong of a regularization on the latent update hurts the performance of the RL agent. Intuitively, the strong regularization could weaken exploration due to a lack of diversity in the representations, thereby hurting performance.

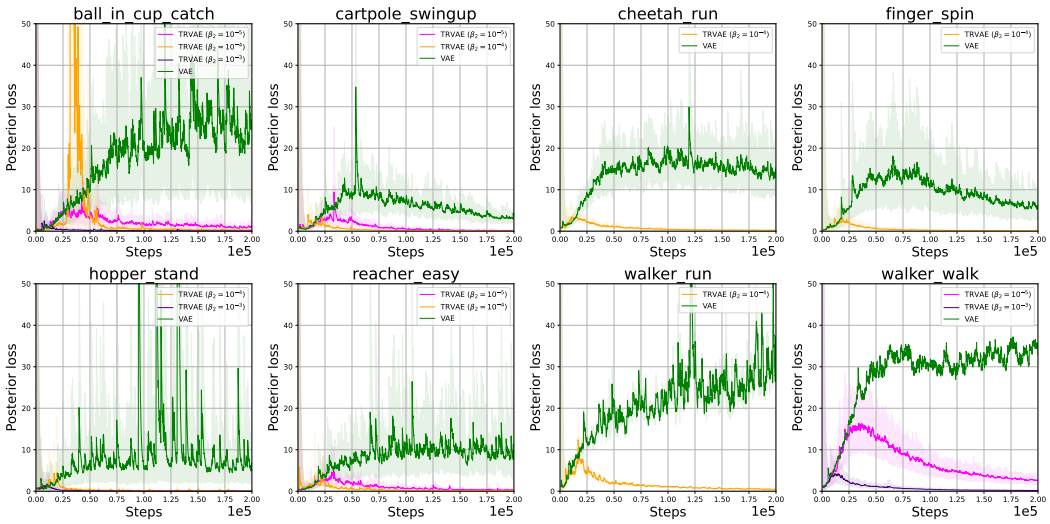


Figure 19: Posterior (trust-region) losses for TRVAE and vanilla VAE across environments. We observe that the posterior loss of the vanilla VAE is significantly higher than the trust-region method. We also see that with a higher β_2 value, the posterior loss is smaller. Smaller posterior losses imply more consistent representations between iterations.

An alternative method for measuring representation drift is to compare the KL divergence in the posteriors of the current and previous autoencoders when encoding the same observation at each timestep. That method maintains invariance in the object of encoding by always encoding the same (e.g., initial) observations. Intuitively, this could provide a more accurate view of the consistency of representation by having the encoder parameters be the only change. However, empirically, both methods yield similar results.