A Derivation of the ELBO

The complete derivation of the ELBO in Equation 1 is as follows.

$$
\begin{aligned}
&\log p\left(\boldsymbol{z}_{1:T}, \boldsymbol{u}_{0:T}\right) \\
=&\log \mathbb{E}_{q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)}\left[\frac{p\left(\hat{s}_{1:T}, \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)}{q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)}\right] \\
\geq&\mathbb{E}_{q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)} \log \left[\frac{p\left(\hat{s}_{1:T}, \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)}{q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)}\right] \\
=&\int q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right) \log \left[\frac{p\left(\hat{s}_{1:T}, \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)}{q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)}\right] d\hat{s}_{1:T} \\
=&\int \sum_{t=1}^{T} q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right) \log \left[\frac{p(\boldsymbol{u}_t \mid \boldsymbol{z}_t)\, p(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1})\, p(\boldsymbol{z}_t \mid \hat{s}_t)}{q_\theta(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}, \boldsymbol{z}_t)}\right] d\hat{s}_{1:T} \\
=&\sum_{t=1}^{T}\left\{\int q_\theta\left(\hat{s}_{1:t} \mid \boldsymbol{u}_{0:t}, \boldsymbol{z}_{1:t}\right) \log \left[p(\boldsymbol{u}_t \mid \boldsymbol{z}_t)\, p(\boldsymbol{z}_t \mid \hat{s}_t)\right] d\hat{s}_{1:t}\right. \\
&\left.+\int q_\theta\left(\hat{s}_{1:t} \mid \boldsymbol{u}_{0:t}, \boldsymbol{z}_{1:t}\right) \log \left[\frac{p(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1})}{q_\theta(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}, \boldsymbol{z}_t)}\right] d\hat{s}_{1:t}\right\} \\
=&\sum_{t=1}^{T}\left\{\int q_\theta\left(\hat{s}_{1:t} \mid \boldsymbol{u}_{0:t}, \boldsymbol{z}_{1:t}\right) \log \left[p(\boldsymbol{u}_t \mid \boldsymbol{z}_t)\, p(\boldsymbol{z}_t \mid \hat{s}_t)\right] d\hat{s}_{1:t}\right. \\
&\left.-\int q_\theta\left(\hat{s}_{1:t-1} \mid \boldsymbol{u}_{0:t-1}, \boldsymbol{z}_{1:t-1}\right) \mathcal{D}_{\mathrm{KL}}\left[q_\theta\left(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}, \boldsymbol{z}_t\right) \| p\left(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}\right)\right] d\hat{s}_{1:t}\right\} \\
=&\mathbb{E}_{q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)} \sum_{t=1}^{T}\left\{\log \left[p(\boldsymbol{u}_t \mid \boldsymbol{z}_t)\right]+\log \left[p(\boldsymbol{z}_t \mid \hat{s}_t)\right]-\mathcal{D}_{\mathrm{KL}}\left[q_\theta\left(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}, \boldsymbol{z}_t\right) \| p\left(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}\right)\right]\right\} \\
\simeq&\sum_{t=1}^{T}\left\{\log \left[p(\boldsymbol{u}_t \mid \boldsymbol{z}_t)\right]+\log \left[p(\boldsymbol{z}_t \mid \hat{s}_t)\right]-\mathcal{D}_{\mathrm{KL}}\left[q_\theta\left(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}, \boldsymbol{z}_t\right) \| p\left(\hat{s}_t \mid \hat{s}_{t-1}, \boldsymbol{u}_{t-1}\right)\right]\right\},
\end{aligned}
\tag{3}
$$

where $\hat{s}_{1:T} \sim q_\theta\left(\hat{s}_{1:T} \mid \boldsymbol{u}_{0:T}, \boldsymbol{z}_{1:T}\right)$ and the inequality in Equation 3 is obtained via Jensen's inequality.

B Experimental Setup

In order to clarify the generalization of our proposed method, we used three different Dec-POMDP domains, SMAC, Google Research Football, and Multi-Agent Discrete MuJoCo, as experimental platforms. All experiments in this paper are carried out with five different seeds on Nvidia GeForce RTX 3090 and Intel(R) Xeon(R) Platinum 8280. We use the official codes for other baseline algorithms, and the hyperparameters are consistent with their original work. Next, we will introduce the settings of the three environments, respectively.

B.1 SMAC

In SMAC, each agent can only obtain entity information within the visible range. The goal of training is to guide the allied agents to defeat the enemy units, so the reward function is related to the health value of all enemy agents. Besides, agents can obtain the current set of available actions. We set the dimension of the latent state in SMAC as the product of 16 and the number of allied agents, and other training hyperparameters for MBVD follow that of QMIX. For MBVD, each independent experiment takes 8 to 30 hours, which is the same as the time spent by QMIX.

We used StarCraft version SC2.4.6.2.69232 instead of the relatively easy version SC2.4.10. The results for different versions are not directly comparable since the underlying dynamics differ. Table 1 provides an overview of the SMAC scenarios. The recognized difficulties of the scenarios are also determined based on the version SC2.4.6.2.69232 of StarCraft.

Name	Ally Units	Enemy Units	Type	Difficulty
2s3z	2 Stalkers 3 Zealots	2 Stalkers 3 Zealots	Heterogeneous Symmetric	Easy
3s5z	3 Stalkers 5 Zealots	3 Stalkers 5 Zealots	Heterogeneous Symmetric	Easy
1c3s5z	1 Colossus 3 Stalkers 5 Zealots	1 Colossus 3 Stalkers 5 Zealots	Heterogeneous Symmetric	Easy
5m_vs_6m	5 Marines	6 Marines	Homogeneous Asymmetric	hard
bane_vs_bane	4 Banelings 20 Zerglings	4 Banelings 20 Zerglings	Heterogeneous Symmetric	hard
2c_vs_64zg	2 Colossi	64 Zerglings	Homogeneous Asymmetric Large Action Space	hard
MMM2	1 Medivac 2 Marauders 7 Marines	1 Medivac 3 Marauder 8 Marines	Heterogeneous Asymmetric Macro tactics	Super Hard
27m_vs_30m	27 Marines	30 Marines	Homogeneous Asymmetric Massive Agents	Super Hard
3s5z_vs_3s6z	3 Stalkers 5 Zealots	3 Stalkers 6 Zealots	Heterogeneous Asymmetric	Super Hard

Table 1: Maps in different scenarios.

B.2 Google Research Football

In Google Research Football, we need to train our players to kick the ball into the opponent's goal. We chose three official scenarios in the Football Academy: *academy_3_vs_1_with_keeper*, *academy_pass_and_shoot_with_keeper*, and *academy_run_pass_and_shoot_with_keeper*. The initial positions of all players and the ball in the three scenarios are shown in Figure 8. We control all the agents in red (except the goalkeeper on the far left) against the agents in blue, which are controlled by built-in AI. Our players have 19 discrete actions, including moving, passing, shooting ,and so on. Unlike SMAC, the agents in Google Research Football cannot know which actions are feasible. The global state of the environment includes the two-dimensional position coordinates and moving directions of all agents on the field, as well as the three-dimensional position coordinates and moving directions of the football. The physical meaning of the local observation is the same as that of the global state, except that the absolute positions of all entities are replaced with relative ones. We also ignore the identifier of agents.

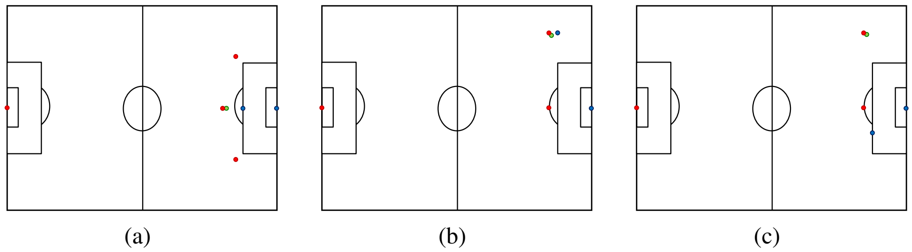

(a)
(b)
(c)

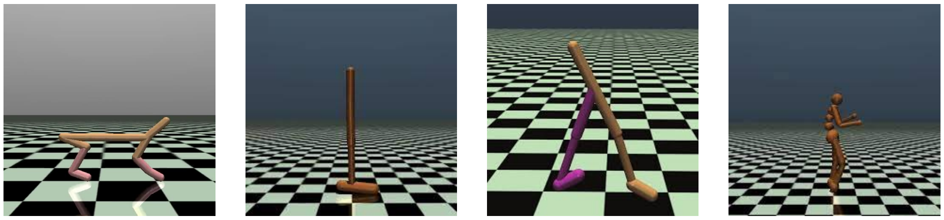
Figure 8: The initial position of each agent in the Google Research Football scenarios considered in our paper: (a) *academy_3_vs_1_with_keeper*, (b) *academy_pass_and_shoot_with_keeper*, and (c) *academy_run_pass_and_shoot_with_keeper*. The red dots represent our players, and the blue dots denote the opposing players. The football is represented by green dots.

For the reward function, in addition to the reward for scoring a goal, we also added an additional reward contribution for moving the ball close to the opponent's goal, similar to the official CHECKPOINT reward function in the Football Engine. To increase the game's difficulty and speed up the agent's training, we set the episode to end when the ball returns to the left half. Besides, scoring goals and reaching the maximum time step will also cause the episode to be terminated.

Due to the small number of agents in Google Research Football scenarios, we adjusted the latent state dimension to the product of 8 and the number of our players. In addition, since there is no information about the feasible action set in this environment, we ignore the $\mathcal{L}_{\text{FA}}$ item in Equation 2. All experiments in Google Research Football were completed within two days.

B.3 Multi-Agent Discrete MuJoCo

In the original Multi-Agent MuJoCo, the action space of each joint is $[-1, 1]$. To accommodate algorithms like QMIX, we discretize the action space into K equally spaced atomic actions. The set of atomic actions for any joint is $\mathcal{A} = \left\{ \frac{2j}{K-1} - 1 \right\}_{j=0}^{K-1}$. We treat each joint as an agent, and joints are connected by adjacent edges. A configurable parameter $l \geq 0$ determines the maximum graph distance to the agent at which joints are observable. The agent observation is then given by a fixed order concatenation of the representation vector of each observable joint. Furthermore, all features in the state are normalized. With the above modifications, we get a benchmark for cooperative multi-agent robotic control with discrete action spaces. In this paper, we set $K = 31$ and $l = 1$.



(a) HalfCheetah (6) (b) Hopper (3) (c) Walker (6) (d) Humanoid (17)

Figure 9: Illustration of benchmark tasks in Multi-Agent MuJoCo. The numbers in brackets mean the number of joints (agents) contained in each robot.

B.4 Hyperparameters

In this paper, we use the QMIX-style framework with its default hyperparameters suggested by the original paper for the reinforcement learning process of MBVD. Table 2 presents the hyperparameters of MBVD. We use n to represent the number of our agents in the environment.

Description	**Value**
Type of optimizer	RMSProp
RMSProp param α	0.99
RMSProp param ϵ	0.00001
Learning rate	0.0005
How many episodes to update target networks	200
Reduce global norm of gradients	10
Batch size	32
Capacity of replay buffer (in episodes)	5000
Discount factor γ	0.99
Starting value for exploraton rate annealing	1
Ending value for exploraton rate annealing	0.05
Horizon of the imagined rollout k	3
KL balancing α	0.3
Dimension of the latent state $\hat{s}$ in SMAC	$n \times 16$
Dimension of the latent state $\hat{s}$ in Google Research Football	$n \times 8$
Dimension of the latent state $\hat{s}$ in Multi-Agent Discrete MuJoCo	$n \times 8$
Dimension of the aggregated rollout state $\hat{s}^{\text{Rollout}}$	Same as that of the real state s

Table 2: Hyperparameter settings.

C Additional Experimental Results

C.1 Results of Other Scenarios in SMAC

We give the performance of all algorithms on other official SMAC maps in Figure 10. The version of StarCraft II in the paper is SC2.4.6.2.69232. The reason why we use this difficult version is that the difficulty of each map is originally delineated according to this version. We do not think there will be a significant change in the ranking of algorithm performance even in SC2.4.10. In *3s_vs_5z* and *corridor*, MBVD performs worse than some other baselines, which we believe is caused by its basic algorithm QMIX. As can be seen from the *3s_vs_5z* scenario, MBVD can still improve the sample efficiency of QMIX. In both *6h_vs_8z* and *corridor* maps, QMIX fails to solve tasks, which is why MBVD performs poorly in both scenarios. However, MBVD based on QMIX performs better than other baselines in most scenarios and can be applied to almost all value decomposition methods to improve the sample efficiency of the original algorithm.

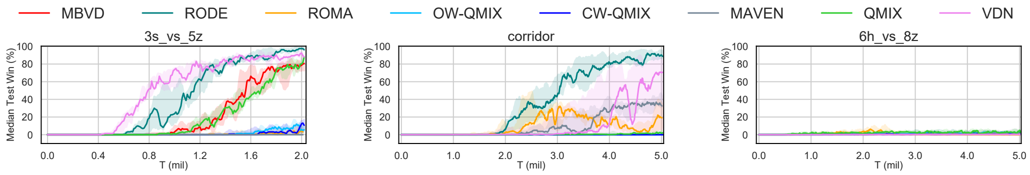


Figure 10: Comparisons between MBVD and baselines on all other maps in SMAC.

C.2 Additional Ablation Studies

We perform ablation studies on other SMAC maps and show the results. The experiments are performed in the easy map *1c3s5z* and the super hard map *MMM2*, respectively. We still explore the role of the imagination module first. In Figure 11, QMIX-RS and QMIX-LS still perform poorly, especially in *MMM2*, which clarifies that the inconsistency between the current policy and the policy that generates imagined states is detrimental to the reinforcement learning process.

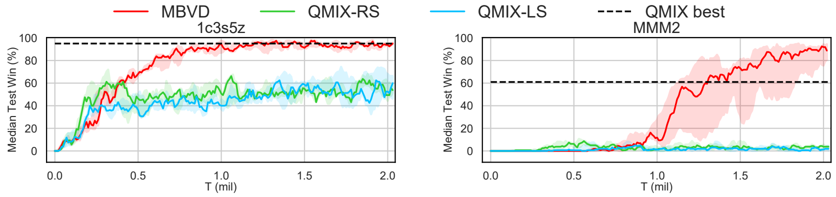


Figure 11: Results for ablation studies on two maps.

Next, we continue to explore how horizon length affects the performance of MBVD. The performance of MBVD under different k values in these two scenarios is shown in Figure 12. In the easy scenario *1c3s5z*, the sample efficiency of MBVD under smaller k values is higher; in the super hard scenario *MMM2*, the opposite is true. Therefore, we conclude that the optimal value of k is different in different scenarios. Longer rollout horizons in easy scenarios will introduce more instability early in training. In hard scenarios, small values of k can make MBVD underutilize its imagination.

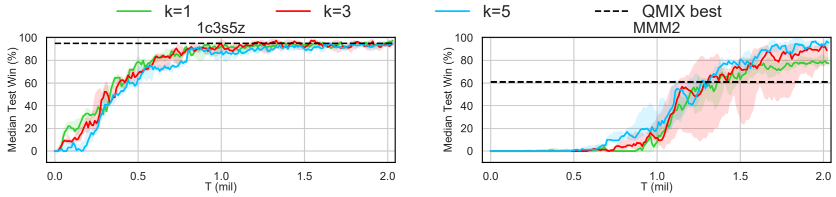


Figure 12: MBVD with different k values on two maps.