MASDIFF: LARGE SCALE MULTI-AGENT SYSTEM EMERGENCE CONTROL VIA EVOLUTIONARY DIFFUSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward model learning methods are primarily divided into implicit reward modeling(IRM) and explicit reward modeling. Implicit reward modeling aims to learn the intrinsic reward of each agent to facilitate better explorationwhile explicit reward modeling(ERM) aims to learn the behavioral preferences of agents. The biggest difference between implicit and explicit reward modeling is that ERM is transferable to other scenarios but IRM cannot. Currently, few methods can simultaneously archive the global target and also learn the ERM. However, the problem addressed in this paper requires the integration of the two objectives. This paper use the diffusion model to generate the expert data for learning ERM. Since the traditional diffusion model can only generate data according to the given expert data, we introduce evolutionary diffusion model to generate data in the absence of any expert data. To steer the collaboration of all agents towards the specified macro-level objective, the macro-level objective is adopted as the fitness for the population. This mechanism transforms the multi-agent exploration based on intrinsic reward into the evolutionary exploration based on genetic operators. Moreover, the optimal individual retention strategy in the evolutionary diffusion model can address the non-stationary problem in MARL. In the experiments, we demonstrate that MASDiff can simultaneously archive the two objectives. Furthermore, we demonstrate that the ability to conduct counterfactual reasoning with the transferable ERM in different scenarios. We propose several 'What if' questions to indicate the change of scenarios and obtain relatively accurate counterfactual reasoning results.

1 INTRODUCTION

Reward model learning methods are primarily divided into implicit reward modeling and explicit reward modeling. Implicit reward modeling aims to learn the intrinsic reward of each agent to facilitate better exploration, ultimately achieving the overall outcomes. Intrinsic reward shaping(Cao et al., 2023), self-supervised RL(Pecháč et al., 2024) and credit assignment(Zhou et al., 2020) can be classified as implicit reward modeling. Explicit reward modeling aims to learn the behavioral preferences of agents by explicitly learning a reward model. The purpose is to ensure that the distribution of behavioral patterns derived from the reward model aligns with the statistical distribution observed in the expert data. Inverse RL(Wang et al., 2024), behavior cloning(Zhou et al., 2024) and Reinforcement Learning with Human Feedback(RLHF)(Casper et al., 2023) can be classified as implicit reward modeling. Currently, there is hardly any method capable of simultaneously achieving the two aforementioned objectives, namely, learning explicit reward models to realize macro objectives in multi-agent systems. This is primarily because existing approaches are mainly designed to address conventional reinforcement learning problems, which do not require the integration of these two objectives. However, the problem addressed in this paper requires the integration of the two objectives, i.e. to learn the explicit reward model for each agent on the one hand and to archive the overall outcome. We first present the specific problem to be solved in this study. Then, we give the contributions of this paper.

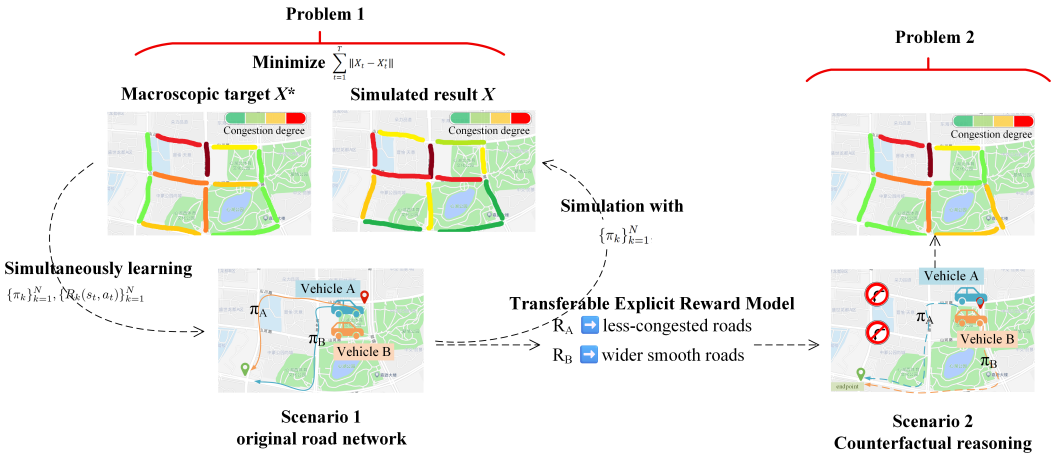


Figure 1: Example of the problems to be addressed by this paper

Consider the following two problems illustrated in Fig 1:

Problem 1: In scenario 1, can we learn the navigation policy of each vehicle such that the simulated congestion conditions aligns with the real congestion status? We assume that the number of vehicles is known and the origin-destination pairs for each vehicle are given.

Problem 2: In scenario 2, what the congestion conditions will be if we prohibit the left turns at two intersections?

If only problem 1 needs to be solved, we can use the implicit reward modeling approach in MARL. This method can learn the navigation policy instead of the explicit reward model for each vehicle to reconstruct the real congestion condition. But in order to solve problem 2 which is more important, the explicit reward model for each vehicle also need to be learned when solving problem 1. Because although the vehicle's optimal route may be different in various scenarios, its decision preference remains the same. For example, vehicle A prefers less-congested roads, while vehicle B prefers wider arterial roads. In scenario 1, vehicle As route is shown as the blue line and vehicle Bs as the orange line. In scenario 2, vehicle A stays on the same route (blue dashed line), while vehicle B shifts to another road (orange dashed line). So to learn the explicit reward model, we need to obtain the expert data for each vehicle in scenario 1 which is transferable in scenario 2. To answer the what-if questions question 2, vehicles simply need to learn the new navigation policy based on the reward model as illustrated in Figure 1. In real world, many issues are similar to problem 1 and 2(Jiaoling et al., 2025). Problem 1 can be generalized as Formula 1.

$$\begin{aligned} \{\pi_k\}_{k=1}^N, \{R_k(s_t,a_t)\}_{k=1}^N &= \text{Argmin} \sum_{t=1}^{T} \|X_t - X_t^*\| \\ &\text{s.t.} \\ \pi_k = \arg\max \mathbb{E}\left[\sum_{t=1}^{T} \gamma^t r_{k,t}\right] \quad &\text{and} \quad r_{k,t} = R_k(s_t,a_t) \end{aligned} \tag{1}$$

We need to learn $\{\pi_k\}_{k=1}^N$ to cooperatively minimize $\sum_{t=1}^{T} \|X_t - X_t^*\|$. And also we need to learn $R_k(s_t, a_t)$ for so that π_k can maximize $\sum_{t=1}^{T} \gamma^t r_{k,t}$. Traditional cooperative MARL methods such as MAPPO(Yu et al., 2022) or QMIX(Rashid et al., 2020) only require $\{\pi_k\}_{k=1}^N$ instead of $R_k(s_t, a_t)$ to be learned. Traditional explicit reward modeling methods such as GAIL(Ho & Ermon, 2016),RLHF(Christiano et al., 2017) only require $R_k(s_t, a_t)$ instead of $\{\pi_k\}_{k=1}^N$ to be learned. When the number of agents is 1000+, problem 1 becomes the emergence control problem. Through interactions of agents guided by microscopic policies, different types of macroscopic properties, such as different types of congestion patterns, may emerge. Although from a research perspective,

1000+ agents pose significant challenges for both algorithmic design and hardware implementation. From a practical perspective, it is highly prevalent. For example, regional traffic networks typically involves 1000+ vehicles. Airport terminals usually accommodate 1000+ pedestrians. Problem 2 is actually the counterfactual reasoning based on the explicit reward model obtained by solving problem 1. To distinguish from the problems typically addressed by existing MARL, we collectively refer to the solutions for problem 1 and 2 as emergence control.

The solution to this problem primarily involves two main challenges: (1)To maximize $\sum_{t=1}^{T} \|X_t - X_t^*\|$ aligns with the goal of implicit reward modeling(Kontogiannis et al., 2025). To explicitly learn the navigation preference $R_k(s_t, a_t)$ for each vehicle k aligns with the goal of explicit reward modeling. However, a simple combination of these two methods cannot effectively solve the integrated problem posed by their superposition. (2) Currently, the primary computational frameworks capable of supporting large-scale multi-agent reinforcement learning (involving 1000+ agents) are population-based MARL and Swarm RL. However, population-based MARL operate on the parameters of policies(Li et al., 2023). It cannot learn an explicit reward model for each individual agent. Swarm RL such as the Mean Field Reinforcement Learning requires the agents in the swarm are interchangeable. Neither of these frameworks can solve the emergence control problem.

We propose the MASDiff training framework that utilizes an evolutionary diffusion model. The evolutionary diffusion model can simultaneously learn the explicit reward model for 1000+ agents and enhance the cooperation of multi-agents to reach the macroscopic target. The contributions of this paper are: **(1)**Since we don't have the expert dataset for training explicit reward model. We use the diffusion model to generate the expert data. **(2)**Since the traditional diffusion model can only generate data according to the given data, we introduce evolutionary diffusion model to generate data in an evolutionary manner. This mechanism dose not require any training data and guarantees that we can obtain the explicit reward model $\{R_k(s_t, a_t)\}_{k=1}^{N}$. **(3)**To steer the collaboration of all agents towards the specified macro-level objective, MASDiff employs the macro-level objective, i.e. $\mathrm{Argmin} \sum_{t=1}^{T} \|X_t - X_t^*\|$ in Formula 1, as the fitness for the evolutionary population. It transforms the multi-agent exploration based on intrinsic reward into the evolutionary exploration based on genetic operators. **(4)**The optimal individual retention strategy in the evolutionary diffusion model can address the non-stationary problem in MARL. **(5)**MASDiff adopt the DTDE and offline-to-online-to-offline training framework to support MARL with 1000+ agents. Since the evolutionary diffusion model can generate the training data for each agent, the learning process of each agent can be executed in parallel by using offline RL for each agent. Training of instance in population can be executed in parallel in an offline manner. Only sample collection are conducted online.

2 Related work

Explicit reward modeling. Current approaches for explicit reward modeling fall into three main categories. **(1)Behavioral Cloning (BC).** BC method learns intelligent body actions directly (Hussein et al., 2017)(Le Mero et al., 2022)(Zhou et al., 2024). It does not require learning reward functions but needs a large amount of expert trajectory data, split into state-policy pairs to represent the experts actions in the current state. **(2)Inverse Reinforcement Learning (IRL).** IRL infers a reward function by modeling expert trajectories, and learning a reward function that emulates expert decision patterns(Adams et al., 2022)(Fernando et al., 2021)(Wang et al., 2024). **(3)Reinforcement Learning with Human Feedback (RLHF).** RLHF aligns agents preferences with real-world behavior by using human feedback to optimize reward functions(Casper et al., 2023)(Kaufmann et al., 2025)(Cao et al., 2023). All three methods require large-scale trajectory datasets, which is problematic in our case due to the lack of granular trajectory data for individual vehicles.

Implicit Reward modeling. Implicit reward modeling aims to learn the intrinsic reward of an agent to facilitate better exploration in multi-agent reinforcement learning, ultimately achieving superior overall outcomes. **(1)Single-Agent Exploration.** Reward bonuses based on the inverse state-action count have been shown to be effective in accelerating learning (Strehl & Littman, 2008). In order to scale count-based approaches to large state spaces, state counts are use as reward bonuses (Ostrovski et al., 2017)(Tang et al., 2017). Some work has focused on defining intrinsic rewards for exploration based on inspiration from psychology (Oudeyer & Kaplan, 2007). **(2)Multi-Agent Exploration.** MAVEN (Mahajan et al., 2019) adopts hierarchical control and the policies of the agents are conditioned on the shared latent variable generated by a hierarchical policy. Wang et al. (2020)

propose two exploration methods, EITI and EDTI, to induce cooperative exploration by capturing the influence of one agents on other agents. CMAE (Liu et al., 2021) proposes that reward function only depends on a small subset of the large state space. EMC (Zheng et al., 2021) proposes that local Q function of each agent can capture the novelty of states and the influences between agents.

Both of the explicit and implicit reward model learning methods are unable to solve our problem. The explanation are given in the above paragraphs.

MARL training architecture for 1000+ agents. Currently, the primary computational frameworks capable of supporting large-scale multi-agent reinforcement learning are population-based MARL and Swarm MARL. **(1)Population-based MARL.** Population-based MARL is a feasible approach to support distributed learning. Prior works include population-based training (Carroll et al., 2019)(Jaderberg et al., 2019), self-play (Vinyals et al., 2019)(Heinrich et al., 2015) and meta-game (Muller et al., 2020)(Omidshafiei et al., 2019). However, population-based MARL cannot learn an explicit reward model for each individual agent. **(2)Swarm MARL.** MARL for swarm system is described as a swarm MDP environment (Hüttenrauch et al., 2019). Among the MARL methods for swarm systems, mean field theory are wildly adopted. Mean Field Reinforcement Learning is proposed to solve the scalability and the interaction of the population of agents(Guo et al., 2023)(Gu et al., 2023)(Anahtarci et al., 2023)(Perrin et al., 2020). However, those methods require two important properties of swarm systems. (1) the agents in the swarm are interchangeable.(2) the exact number of agents in the swarm is irrelevant. It also cannot learn an explicit reward model for each individual agent.

Diffusion models for reinforcement learning. Offline reinforcement learning is a powerful tool for addressing the problem of action sequence combination explosion. However, its performance is ultimately constrained by the offline dataset. Many studies have introduced diffusion models into reinforcement learning(Kang et al., 2023),(Hansen-Estruch et al., 2023),(Ren et al., 2025) . Zhu et al. (2023) summarizes the application of diffusion models in reinforcement learning-related fields. Janner et al. (2022) adopts diffusion models as planners to replace traditional autoregressive-based planners. SafeDiffuser (Xiao et al., 2025) introduces safety constraints to generate safe trajectories. Diffusion-QL (Wang et al., 2023) integrates diffusion policies into the Q-learning framework to address the distribution shift issue in offline RL. CEP (Lu et al., 2023) uses contrastive performance prediction to guide energy function sampling. MADiff (Zhu et al., 2024) uses an attention-based diffusion model to model the complex coordination among behaviors of multiple agents. All those methods require training data and thus is not applicable for solvin our problem.

3 METHODOLOGY

3.1 IDEAL TRAINING FRAMEWORK VIA ORACLE

We define a Markov Decision Process as (S, A, R, $\boldsymbol{\pi}$), where $S = \{s_1, s_2, ..., s_N\}$, $A = \{a_1, a_2, ..., a_N\}$ and $R = \{r_1, r_2, ..., r_N\}$ are the joint set of individual states, actions and rewards, respectively. Once an agent reaches its next state, it receives a reward r. $\boldsymbol{\pi} = \{\pi_i\}_{i=1}^N$ are the combination of policies. Let $\tau_{\pi_i} = \{(s, a, s')_t\}_{t=1}^T$ denote trajectory sampled by π_i where $s, s' \in S_i$ and $a \in A_i$. Let $\boldsymbol{\tau} = \{\tau_{\pi_i}\}_{i=1}^N, \boldsymbol{\tau} \in \mathbb{R}^{|S|*|a|*|S|*|T|*|N|}$ denote the combined trajectory collected by $\{\pi_i\}_{i=1}^N$. Let $R = \{\boldsymbol{r}_i\}_{i=1}^N$, $\boldsymbol{r}_i = \{r_t\}_{t=1}^T$, $\boldsymbol{r}_i$ denotes the corresponding rewards for $\tau_{\pi_i} \in \boldsymbol{\tau}$.

Each $\pi_i \in \boldsymbol{\pi}$ can be obtained by conducting offline reinforcement learning on $\{\tau_{\pi_i} \cup \boldsymbol{r}_i\} \in \{\boldsymbol{\tau} \cup R\}$ via the DTDE framework. Let $\{\{X_t\}_{t=1}^T\}_{\{\boldsymbol{\tau} \cup R\}}$ denotes the macroscopic status by conducting simulation on $\boldsymbol{\pi}$ deriving from $\{\boldsymbol{\tau} \cup R\}$. Let $\{X_t^*\}_{t=1}^T$ denotes the target macroscopic state. If we can obtain the optimal $\{\boldsymbol{\tau} \cup R\}$, then the optimal $\boldsymbol{\pi}$ can be derived. This paper firstly designs an ideal training framework based on an oracle $\mathcal{O}$ to obtain the optimal $\{\boldsymbol{\tau} \cup R\}$.

Oracle reward model. We denote by $\mathcal{O}(\boldsymbol{\tau})$ an oracle reward model satisfying the following guarantee for $\boldsymbol{\tau}$. For input $\boldsymbol{\tau}$, the oracle outputs $R \leftarrow \mathcal{O}(\boldsymbol{\tau})$. For any other R', $R' \neq R$, we have $\sum_{t=1}^T ||X_t - X_t^*|| \leq \sum_{t=1}^T ||X_t' - X_t^*||$, where $X_t \in \{\{X_t\}_{t=1}^T\}_{\{\boldsymbol{\tau} \cup R\}}$ and $X'_t \in \{\{X'_t\}_{t=1}^T\}_{\{\boldsymbol{\tau} \cup R'\}}$.

Proposition 1. $\sum_{t=1}^T ||X_t - X_t^*||$ is monotonically decreasing in each iteration of Algorithm 1.

Algorithm 1 Ideal training framework

1: Input: Oracle reward model $\mathcal{O}$, target macro emergent phenomenon $\{X_t\}_{t=1}^T$
2: Initializing policy combination $\boldsymbol{\pi} = \{\pi_i\}_{i=1}^N$
3: **while** $\sum_{t=1}^T ||X_t - X_t^*||$ not converged **do**
4: Conduct simulation with $\boldsymbol{\pi} = \{\pi_i\}_{i=1}^N$, obtaining $\boldsymbol{\tau}$ and $\{X_t\}_{t=1}^T$
5: Query Oracle $\mathcal{O}$ to obtain R, $R \leftarrow \mathcal{O}(\boldsymbol{\tau})$
6: Update $\boldsymbol{\pi} = \{\pi_i\}_{i=1}^N$ by offline RL via DTDE framework with $\{\boldsymbol{\tau} \cup R\}$
7: **end while**

Proof. Based on oracle reward model $\mathcal{O}$, the MARL in Algorithm 1 is actually transformed into a single-agent reinforcement learning. $\{X_i\}_{t=1}^T$ can be viewed as the ordinary reward value. The entire process is equivalent to classical policy iteration, which has been proven to yield monotonic improvement in returns, corresponding to a monotonic decrease of $\{X_i\}_{t=1}^T$ in Algorithm 1.

Based on Proposition 1, we introduce the MASDiff framework that use a diffusion model to approximate oracle $\mathcal{O}$. MASDiff shift the focus from exploring the optimal $\boldsymbol{\pi}$ to exploring the optimal R on condition of $\boldsymbol{\tau}$. In the article "Reward is enough" (Silver et al., 2021), the author believes rewards are sufficient to drive behaviors. Therefore, the diffusion model essentially generates the decision preferences for agents.

3.2 Approximating oracle $\mathcal{O}$ with diffusion model

To approximate oracle $\mathcal{O}$, the diffusion model is to generate R on condition of $\boldsymbol{\tau}$. From a practical perspective, it is also necessary to treat $\boldsymbol{\tau}$ as a condition obtained from the environment. Because the generated $\boldsymbol{\tau}$ may be inconsistent with reality. This paper adopts the denoising diffusion probability model (Ho et al., 2020) as the basic diffusion model. We firstly give the description of the basic diffusion model. Then, we describe how to train the model such that it can approximate $\mathcal{O}$.

3.2.1 Conditional denoising diffusion probability model

To generate R by incorporating $\boldsymbol{\tau}$ as conditions, the original distribution R_0 of the diffusion process can be transformed into an isotropic Gaussian distribution $\mathcal{N}(0,1)$ through T steps of forward diffusion. The diffusion model denoted as p_θ attempts to recover the original distribution of R_0 under the condition of $\boldsymbol{\tau}$. Moreover, the dimension of $\boldsymbol{\tau}$ is $\mathbb{R}^{|S|*|a|*|S|*|T|*|N|}$ which is quite high when N>1000 and T>100. We reduce the dimensions of $\boldsymbol{\tau}$ and R with encoders by mapping $\boldsymbol{\tau}$ and R to latent space representation $Z_{\boldsymbol{\tau}}$ and Z_R.

The forward process is parameterized as:

$$q\left(Z_{R1:T} \mid Z_{R0}, Z_{\boldsymbol{\tau}}\right) = \prod_{t=1}^{T} q\left(Z_{Rt} \mid Z_{Rt-1}, Z_{\boldsymbol{\tau}}\right) \tag{2}$$

$$q\left(Z_{Rt} \mid Z_{Rt-1}, Z_{\boldsymbol{\tau}}\right) = \mathcal{N}\left(Z_{Rt}; \sqrt{1-\beta_t} Z_{Rt-1}, \beta_t I\right) \tag{3}$$

where $q\left(Z_{Rt} \mid Z_{Rt-1}, Z_{\boldsymbol{\tau}}\right)$ represents the conditional Gaussian distribution, and the variance β_t adjusts the noise level. Z_{Rt} at any time step t can be directly represented by Z_{R0} as:

$$Z_{Rt} = \sqrt{\alpha_t} Z_{R0} + \sqrt{1-\alpha_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0,1) \tag{4}$$

$$\alpha_t = \prod_{s=1}^{T}(1-\beta_s) \tag{5}$$

The backward process is parameterized as:

$$p_\theta\left(Z_{R0}, \ldots, Z_{Rt-1} \mid Z_{RT}, Z_{\boldsymbol{\tau}}\right) = p_\theta\left(Z_{RT}\right) \prod_{t=1}^{T} p_\theta\left(Z_{Rt-1} \mid Z_{Rt}, Z_{\boldsymbol{\tau}}\right) \tag{6}$$

where $Z_{RT} \sim \mathcal{N}(0,1)$ and $p_\theta\left(Z_{Rt-1} \mid Z_{Rt}, Z_{\boldsymbol{\tau}}\right)$ is assumed to follow a normal distribution with learnable parameters. The backward process can be trained by optimizing the following objective:

$$L = \min_\theta \mathbb{E}_{Z_{Rt}, Z_{\boldsymbol{\tau}}, \epsilon, t}\left[\left\|\epsilon - \epsilon_\theta\left(Z_{Rt}, Z_{\boldsymbol{\tau}}, t\right)\right\|_2^2\right] \tag{7}$$

where $\epsilon_\theta\left(Z_{Rt}, Z_{\boldsymbol{\tau}}, t\right)$ predicts the noise using a neural network.

3.2.2 TRAINING OF DIFFUSION MODEL

Training of diffusion model implies two layers of meaning. Firstly, let $(\{\boldsymbol{\tau} \cup R\}, \rho(\{\boldsymbol{\tau} \cup R\}))$ denote a training sample for p_θ. $\rho(\{\boldsymbol{\tau} \cup R\}) = \sum_{t=1}^{T} ||X_t - X_t^*||$ denotes the weight of $\{\boldsymbol{\tau} \cup R\}$, where $X_t \in \{\{X_t\}_{t=1}^T\}_{\{\boldsymbol{\tau} \cup R\}}$. $\rho(\{\boldsymbol{\tau} \cup R\})$ is normalized based on Formula 8. $\min_\rho$ and $\max_\rho$ denotes the minimum and maximum value of ρ in the sample set. To obtain high-quality samples, γ is further applied for scaling. If we have enough samples, we can train the model according to Formula 9 (Kumar & Levine, 2020). The weighted training can let the diffusion model generate sample with higher ρ values.

$$\rho(\{\boldsymbol{\tau} \cup R\}) = \left(1 - \frac{\rho - \min_\rho}{\max_\rho - \min_\rho}\right) * \gamma \tag{8}$$

$$L = \min_\theta \mathbb{E}_{R_t, \boldsymbol{\tau}, \epsilon, t}\left[\rho(\{\boldsymbol{\tau} \cup R\}) * \left\|\epsilon - \epsilon_\theta\left(R_t, \boldsymbol{\tau}, t\right)\right\|_2^2\right] \tag{9}$$

Secondly, since there is not any samples at the beginning, we need to train p_θ in a bootstrapping manner. In other words, p_θ need to generate training sample for itself. Moreover, p_θ need to explore samples that have higher ρ value. We design the evolution mechanism to let p_θ explore better new samples. The evolutionary strategy consists of four steps illustrated as follows.

Population. Let $\mathcal{T} = \{\{\boldsymbol{\tau} \cup R\}_j\}_{j=1}^M$ denote the population of $\{\boldsymbol{\tau} \cup R\}$. $\mathcal{T}$ consists M instances of $\{\boldsymbol{\tau} \cup R\}$ where $R \sim p_\theta(R|\boldsymbol{\tau})$. Let $\Pi = \{\boldsymbol{\pi}_j\}_{j=1}^M$,$\boldsymbol{\pi} = \{\pi_i\}_{i=1}^N$ denote the policy population. Π consists M instances of $\boldsymbol{\pi}$. We randomly initialize the policy population Π. The initial population of $\mathcal{T}$ is the rollout data by Π.

Sample scoring. At each iteration k, we score each $\{\boldsymbol{\tau} \cup R\} \in \mathcal{T}^k$ with $\rho(\{\boldsymbol{\tau} \cup R\})$.

Selection. We use $\rho(\{\boldsymbol{\tau} \cup R\})$ to decide which samples in $\mathcal{T}^k$ should be selected to propagate to the next iteration. Let E^k denotes the elite set selected from $\mathcal{T}^k$ according to the roulettewheel strategy (Lipowska, 2012).

Mutation using truncated diffusion-denoising. We apply randomized mutations to E^k for exploration. A truncated diffusion-denoising process is adopted to mutate trajectories. We run the first t steps of the forward diffusion process to add noise to elite trajectories in E^k based on Formula 10.

$$E_{noisy}^{k+1} = \{\boldsymbol{\tau} \cup R_{noisy}\}, R_{noisy} = \{\sqrt{\alpha_t}R + \sqrt{1-\alpha_t}\epsilon, \epsilon \sim N(0,1) | \{\boldsymbol{\tau} \cup R\} \in E^k\} \tag{10}$$

Then we run the last t steps of the reverse diffusion process to denoise E_{noisy}^{k+1} to obtain clean E_{clean}^{k+1} based on Formula 11. Finally, E_{clean}^{k+1} is the next generation of $\mathcal{T}^k$, i.e. $\mathcal{T}^{k+1}$.

$$E_{clean}^{k+1} = \{\boldsymbol{\tau} \cup R_{clean}\}, R_{clean} \sim \{p_\theta(R_{noisy}|\boldsymbol{\tau}) | \{\boldsymbol{\tau} \cup R_{noisy}\} \in E_{noisy}^{k+1}\} \tag{11}$$

Notation. Both the diffusion model and $\{X_t\}_{t=1}^T$ incorporate T. T denotes the diffusion steps when describing the diffusion model, whereas T represents the time-series length when referring to $\{X_t\}_{t=1}^T$. In Formula 10 and 11, we use R and $\boldsymbol{\tau}$ to denote Z_R and $Z_{\boldsymbol{\tau}}$ for brevity.

3.3 TRAINING FRAMEWORK OF MASDIFF TO SUPPORT MARL WITH 1000+ AGENTS

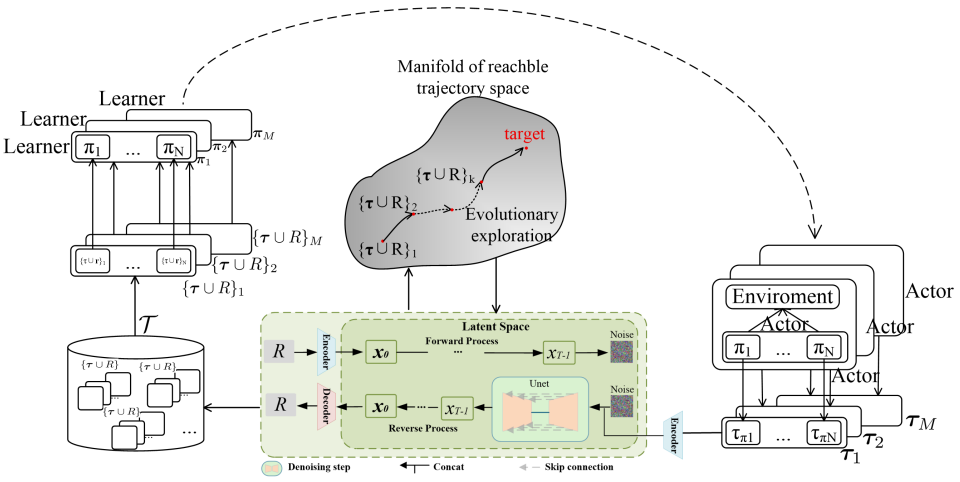


Figure 2: Training framework of MASDiff

Training framework of MASDiff. Algorithm 2 illustrates how to transform the ideal training framework into the practically feasible training framework MASDiff. MASDiff utilizes policy population and sample population to leverage evolutionary search over the trajectory manifold.

Algorithm 2 Training framework of MASDiff

1: Input:Target macro emergent phenomenon $\{X_t\}_{t=1}^T$. number of agents N
2: Initializing policy population Π,$\Pi = \{\boldsymbol{\pi}_j\}_{j=1}^M$,$\boldsymbol{\pi} = \{\pi_i\}_{i=1}^N$
3: Initializing diffusion model p_θ
4: **while** $\sum_{t=1}^T ||X_t - X_t^*||$ not converged or max iteration not reached **do**
5: Simulating with $\Pi = \{\boldsymbol{\pi}_j\}_{j=1}^M$, obtaining $\{\boldsymbol{\tau}\}_{j=1}^M$ for each $\boldsymbol{\pi}_j$ ▷ collect trajectories online
6: Query p_θ to obtain R_j for each $\boldsymbol{\tau}_j \in \{\boldsymbol{\tau}\}_{j=1}^M$, $R_j \sim p_\theta(R_j|\boldsymbol{\tau}_j)$
7: Update each $\boldsymbol{\pi}_j \in \Pi$ by independent offline RL with $\{\boldsymbol{\tau}_j \cup R_j\}$ ▷ update $\boldsymbol{\pi}_j \in \Pi$ offline
8: Simulating with $\Pi = \{\boldsymbol{\pi}_j\}_{j=1}^M$, obtaining $\{\{X_t\}_{t=1}^T\}_j\}_{j=1}^M$ for each $\boldsymbol{\pi}_j$ ▷ collect simulation result for trajectories in line 5 online
9: Update p_θ with weighted sample $\{\{\boldsymbol{\tau}_j \cup R_j\}, \rho(\{\boldsymbol{\tau}_j \cup R_j\})\}_{j=1}^M$
10: Evolving the sample population $\{\boldsymbol{\tau}_j \cup R_j\}_{j=1}^M$ to the next generation $\{\boldsymbol{\tau}_j \cup R'_j\}_{j=1}^M$
11: Update each $\boldsymbol{\pi}_j \in \Pi$ by independent offline RL with $\{\boldsymbol{\tau}_j \cup R'_j\}$ ▷ update $\boldsymbol{\pi}_j \in \Pi$ offline
12: Simulating with $\Pi = \{\boldsymbol{\pi}_j\}_{j=1}^M$, obtaining $\{\{X_t\}_{t=1}^T\}_j\}_{j=1}^M$ for each $\boldsymbol{\pi}_j$ ▷ collect simulation result for trajectories in line 10 online
13: Update p_θ with weighted sample $\{\{\boldsymbol{\tau}_j \cup R'_j\}, \rho(\{\boldsymbol{\tau}_j \cup R'_j\})\}_{j=1}^M$
14: **end while**

In line 5, we use the policy population to collect trajectories. In line 6, rewards are generated by query the diffusion model with the trajectories. In line 7∼9, we update the diffusion model by samples collected in line 5∼6. In line 10∼13, the diffusion model are further updated by samples collected with evolutionary strategy. MASDiff also adopt the offline-to-online-to-offline training framework. Training are all conducted offline. Only sample collection and simulation result collection are conducted online.

Recovering explicit reward model $R_k(s_t, a_t)$ from weighted sample $\{\{\boldsymbol{\tau}_j \cup R'_j\}, \rho(\{\boldsymbol{\tau}_j \cup R'_j\})\}_{j=1}^M$. Assume $\{\{\boldsymbol{\tau}_j \cup R'_j\}, \rho(\{\boldsymbol{\tau}_j \cup R'_j\})\}$ has the highest ρ value. Assume $\{\tau_{\pi_i} \cup \boldsymbol{r}_k\} \in \{\{\boldsymbol{\tau}_j \cup R'_j\}, \tau_{\pi_k} = \{(s, a, s')_t\}_{t=1}^T$ and $\boldsymbol{r}_k = \{r_t\}_{t=1}^T$. We can parameterize $R_k(s_t, a_t)$ using a single-time-step neural network $r_t = g_\theta(s_t, a_t)$ to predict a reward for a state-action pair (s_t, a_t).

Parallel training paradigm for MASDiff. Although the dimension of the trajectory space is $\mathbb{R}^{|S|*|a|*|S|*|T|*|N|}$ which is quite large. The reachable trajectory space is a low dimensional manifold embedded in $\mathbb{R}^{|S|*|a|*|S|*|T|*|N|}$. MASDiff is essentially to efficiently explore the reachable trajectory manifold that can emerge the target macroscopic phenomenon. Figure 2 illustrates the training framework of MASDiff. It decouples the training and rollout tasks using the Actor-Diffusion-Learner model. The Learner operates policy training and the Actor operates data collecting. The diffusion model interleaves the Learners and Actors by providing better samples to the Learner instead of those collected directly by Actors.

To conduct fully decentralized multi-agent learning, MASDiff uses $\tau_{\pi_i} \in \boldsymbol{\tau}$ to train $\pi_i \in \boldsymbol{\pi}$ independently. One learner for one policy, policy learning is independent to other agents. Actor is to collect the rollout data $\{\tau_{\pi_i}\}_{i=1}^{N}$ using $\{\pi_i\}_{i=1}^{N}$. Multiple Actors can be dispatched in parallel. The Actor-Diffusion-Learner architecture are built on top of Ray which allows tasks to be distributed over a large cluster.

4 EXPERIMENT

The following three research questions illustrate the purpose of the experiment:

RQ1: Can MASDiff minimize $\sum_{t=1}^{T} \|X_t - X_t^*\|$?

RQ2: Can the explicit reward model $R_k(s_t, a_t)$ learned by MASDiff answer the 'What if' counterfactual questions in new scenarios?

RQ3: Since the search space of multi-agent reinforcement learning grows exponentially as the number of agents increases, how is the scalability of the MASDiff framework?

4.1 EXPERIMENTAL SETTINGS

Comparison methods. Since existing CTDE or DTDE MARL framework can not work on such large number of vehicles with local reward completely unknown. We design methods for comparison by change or remove part of the algorithms in MASDiff framework.

CGAN replaces the conditional denoising diffusion probabilistic model in MASDiff with Conditional Generative Adversarial Network.

LDM_IMIT adopts imitation learning instead of offline DQN in learning vehicle navigation policy.

LDM_CFG use classifier free guidance in the conditional diffusion model. It incorporates $\rho(\{\tau \cup R\})$ as an additional condition for the diffusion model, i.e. $R \sim p_\theta(R|\boldsymbol{\tau}_j, \rho(\{\tau \cup R\}))$. Instead of utilizing evolutionary strategy to explore new sample, LDM_CFG uses the additional condition $\rho(\{\tau \cup R\})$ to explore new sample.

LDM_PCFG add a proxy in LDM_CFG. The proxy uses $\sum_{t=1}^{T} \|X_t - X_t^*\|$ to predict $\rho(\{\tau \cup R\})$. The condition $\rho(\{\tau \cup R\})$ in LDM_PCFG is given by the proxy in stead of the real value of $\rho(\{\tau \cup R\})$.

Experimental environment. Experiments were conducted on a high-performance server equipped with an AMD EPYC 9554 CPU, four NVIDIA RTX 4090 GPUs (24GB each), 256GB of DDR5 RAM, and a 2TB NVMe SSD. The operation system is Ubuntu 22.04 LTS. The software stack include Python 3.10, the SUMO(Simulation of Urban MObility) simulation environment, and the Ray framework for parallel execution. The population size in Algorithm 2 is 500. The traffic network in 3 which is part of a certain district of Chongqing. This region includes 114 roads and 45 intersections.

4.2 RQ1: CAN MASDIFF MINIMIZE $\sum_{t=1}^{T} \|X_t - X_t^*\|$?

Experimental design. This experiment is conducted in scenario 1 illustrated in Figure 3(b). Figure 3(a) is the original Baidu map. Figure 3(b) is the SUMO traffic road network derived from Figure 3(a). The surveillance camera data from 8:30 to 9:00 am on August 21, 2023 in this region are used for this experiment. There are totally 1557 vehicles in the camera data. We adopt the largest functional cluster of the traffic network (LCC) as the emergent macro traffic state denoted as $\{LCC_t^*\}_{t=1}^{T}$. Functional cluster is the connected roads with relatively high speed (Zeng et al.,

2020). We calculate LCC using the velocity of the 1557 vehicles recorded by surveillance camera. LCC is calculated every minute and thus $T = 30$ in $\{LCC_t^*\}_{t=1}^T$. MASDiff learns the navigation policy for each vehicle according to $\{LCC_t^*\}_{t=1}^{30}$. $\{LCC_t\}_{t=1}^{30}$ is obtained by conduct simulation with the learned policies.

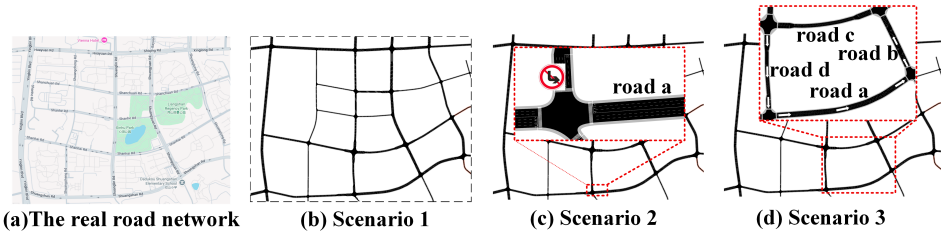


Figure 3: Different experiment scenarios

Experimental results. The results are shown in Figure 4. The horizontal axis represents the iteration times. The vertical axis represents the minimal value of $\|\{LCC_t\}_{t=1}^{30} - \{LCC_t^*\}_{t=1}^{30}\|$ during the past iterations. All methods are iterated 100 rounds. MASDiff obtain the best results. LDM_CFG and LDM_PCFG produce the worst results. This indicates that directly using $\sum_{t=1}^{T} \|X_t - X_t^*\|$ as a condition provides limited guidance for the diffusion model's exploration.

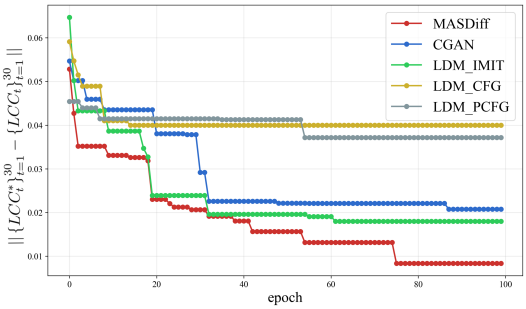


Figure 4: Minimal value of $\| \{LCC_t^*\}_{t=1}^{30} - \{LCC_t\}_{t=1}^{30} \|$ per iteration.

Figure 5 shows the detail of $\{LCC_t\}_{t=1}^{30}$ in each minute. The orange curve is $\{LCC_t^*\}_{t=1}^{30}$. The blue curve is $\{LCC_t\}_{t=1}^{30}$. The curve by MASDiff best approximates the ground truth.

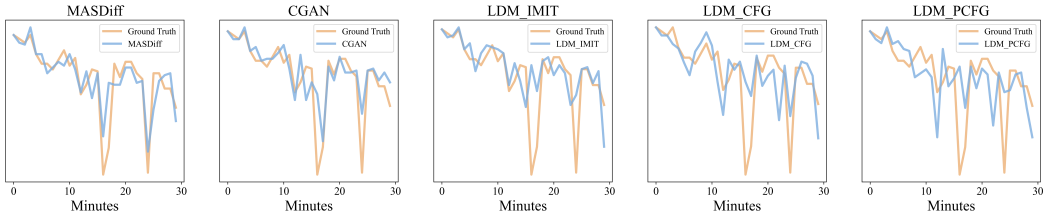


Figure 5: $\{LCC_t^*\}_{t=1}^{30}$ and $\{LCC_t\}_{t=1}^{30}$ in scenario 1.

4.3 RQ2: Can we conduct counterfactual reasoning by the learned explicit reward model?

Experimental design. This experiment is conducted in scenario 2∼3 illustrated in Figure 3(c)∼(d).

Scenario 2: What is the $\{MQL_t\}_{t=1}^{T}$ if we prohibiting the left turns onto road a at the intersections?

Scenario 3: What is the $\{MQL_t\}_{t=1}^{T}$ if we establish a one-way traffic loop in the specific area?

MQL is adopted as the macro emergent traffic state which is the maximum queue length of a road segment in a certain period. $\{MQL_t^*\}_{t=1}^{30}$ represents the ground truth data. $\{MQL_t^*\}_{t=1,s1}^{30} \sim \{MQL_t\}_{t=1,s3}^{30}$ denote the ground truth data in scenario $1 \sim 3$. $\{MQL_t\}_{t=1,s1}^{30} \sim \{MQL_t^*\}_{t=1,s3}^{30}$ denote the results obtained through different methods in scenario $1 \sim 3$.

Due to the change of scenario, $\{MQL_t^*\}_{t=1,s2}^{30}$ and $\{MQL_t^*\}_{t=1,s3}^{30}$ cannot be obtained from the original surveillance camera data or be predicted based on historical data. Therefore, we synthesize the ground truth data through randomly simulated reward models. First, we randomly generate the ground truth reward model for those 1557 vehicles. Based on the generated reward models, we conduct distributed RL for each vehicle in scenario $1 \sim 3$ to obtain the navigation policies. By simulating with the policies we can obtain $\{MQL_t^*\}_{t=1,s2}^{30}, \{MQL_t^*\}_{t=1,s3}^{30}$. To obtain $\{MQL_t\}_{t=1,s2}^{30}, \{MQL_t\}_{t=1,s3}^{30}$, We first learn the explicit reward model and the navigation policy according to $\{MQL_t^*\}_{t=1,s1}^{30}$. Then, $\{MQL_t\}_{t=1,s2}^{30}$ and $\{MQL_t\}_{t=1,s3}^{30}$ can be obtained by conduct simulation with the learned policies in scenario 2 and 3.

Experimental results. In scenario 1, roads a and b are congested. Roads c and d experience lower traffic volumes. In scenario 2, congestion on road a is partially alleviated. In scenario 3, congestion on both roads a and b is reduced. Figures 7, 8 and 9 show heatmaps of $\{MQL_t\}_{t=1,s1}^{30} \sim \{MQL_t\}_{t=1,s2}^{30}$ for each method in different scenarios. The horizontal axis represents the minutes. The vertical axis represents the MQL value of a specific road in each minute. The top 50 roads with the highest MQL in the original scenario were selected. Due to space constraints, we place Figures 7, 8 and 9 in the appendix. In different scenarios, $\{MQL_t\}_{t=1,s1}^{30} \sim \{MQL_t\}_{t=1,s4}^{30}$ by MASDiff are more consistent with $\{MQL_t^*\}_{t=1,s1}^{30} \sim \{MQL_t^*\}_{t=1,s3}^{30}$ than other methods. This means that the learned policies can be used to conduct counterfactual reasoning in hypothetical scenarios.

4.4 RQ3: How is the scalability of the MASDiff framework

Scenario 1 is chose as the simulation environment. We increase the number of vehicles from 1,557 to 2,578 and 4,423 at time intervals of 0.5, 1, and 1.5 hours from 8:30 am. We adopt MQL as the macro metric. The population size is also 500. The results are shown in Tables 1 and 2. In Table 2, running time denotes the time elapsed when the algorithm reaches convergence. The experimental results demonstrates that MASDiff significantly outperforms other methods both in performance and running time. The results obtained by LDM_CFG and LDM_PCFG are nearly identical to that from random sampling.

Table 1: $\|\{MQL_t^*\}_{t=1}^{T} - \{MQL_t\}_{t=1}^{T}\|$

	1557	**2758**	**4423**
MASDiff	1.69	2.19	2.46
CGAN	1.76	5.32	7.24
LDM_IMIT	2.02	2.40	2.91
LDM_CFG	–	–	–
LDM_PCFG	–	–	–

Table 2: Running time(minute)

	1557	**2758**	**4423**
MASDiff	101.25	139.17	218.33
CGAN	172.08	245.83	259.17
LDM_IMIT	113.75	123.33	140.83
LDM_CFG	–	–	–
LDM_PCFG	–	–	–

5 Conclusion

This paper tries to control the emergence of large scale multi-agent system. To solve this problem, the MASDiff framework are proposed. The experimental results verifies that MASDiff can achieve emergence control fairly well. To our best knowledge, MASDiff is the first framework targeting at controlling emergence in MAS. In future work, each part of the framework will be optimized.

REFERENCES

Stephen C. Adams, Tyler Cody, and Peter A. Beling. A survey of inverse reinforcement learning. *Artif. Intell. Rev.*, 55(6):4307–4346, 2022. doi: 10.1007/S10462-021-10108-X. URL https://doi.org/10.1007/s10462-021-10108-x.

Berkay Anahtarci, Can Deha Kariksiz, and Naci Saldi. Q-learning in regularized mean-field games. *Dyn. Games Appl.*, 13(1):89–117, 2023. doi: 10.1007/S13235-022-00450-2. URL https://doi.org/10.1007/s13235-022-00450-2.

Yulong Cao, Boris Ivanovic, Chaowei Xiao, and Marco Pavone. Reinforcement learning with human feedback for realistic traffic simulation. *CoRR*, abs/2309.00709, 2023. doi: 10.48550/ARXIV.2309.00709. URL https://doi.org/10.48550/arXiv.2309.00709.

Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, Sanjit A. Seshia, P. Abbeel, and Anca D. Dragan. On the utility of learning about humans for human-ai coordination. *ArXiv*, abs/1910.05789, 2019. URL https://api.semanticscholar.org/CorpusID:202770731.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, Tony Tong Wang, Samuel Marks, Charbel-Raphaël Ségerie, Micah Carroll, Andi Peng, Phillip J. K. Christoffersen, Mehul Damani, Stewart Slocum, Usman Anwar, Anand Siththaranjan, Max Nadeau, Eric J. Michaud, Jacob Pfau, Dmitrii Krasheninnikov, Xin Chen, Lauro Langosco, Peter Hase, Erdem Biyik, Anca D. Dragan, David Krueger, Dorsa Sadigh, and Dylan Hadfield-Menell. Open problems and fundamental limitations of reinforcement learning from human feedback. *CoRR*, abs/2307.15217, 2023. doi: 10.48550/ARXIV.2307.15217. URL https://doi.org/10.48550/arXiv.2307.15217.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. Deep inverse reinforcement learning for behavior prediction in autonomous driving: Accurate forecasts of vehicle motion. *IEEE Signal Process. Mag.*, 38(1):87–96, 2021. doi: 10.1109/MSP.2020.2988287. URL https://doi.org/10.1109/MSP.2020.2988287.

Haotian Gu, Xin Guo, Xiaoli Wei, and Renyuan Xu. Dynamic programming principles for mean-field controls with learning. *Oper. Res.*, 71(4):1040–1054, 2023. doi: 10.1287/OPRE.2022.2395. URL https://doi.org/10.1287/opre.2022.2395.

Xin Guo, Anran Hu, Renyuan Xu, and Junzi Zhang. A general framework for learning mean-field games. *Math. Oper. Res.*, 48(2):656–686, 2023. doi: 10.1287/MOOR.2022.1274. URL https://doi.org/10.1287/moor.2022.1274.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. IDQL: implicit q-learning as an actor-critic method with diffusion policies. *CoRR*, abs/2304.10573, 2023. doi: 10.48550/ARXIV.2304.10573. URL https://doi.org/10.48550/arXiv.2304.10573.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 805813. JMLR.org, 2015.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Comput. Surv.*, 50(2):21:1–21:35, 2017. doi: 10.1145/3054912. URL https://doi.org/10.1145/3054912.

Maximilian Hüttenrauch, Adrian Sosic, and Gerhard Neumann. Deep reinforcement learning for swarm systems. *J. Mach. Learn. Res.*, 20:54:1–54:31, 2019. URL https://jmlr.org/papers/v20/18-476.html.

Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, and Thore Graepel. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019. doi: 10.1126/science.aau6249. URL https://www.science.org/doi/abs/10.1126/science.aau6249.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 2022. URL https://proceedings.mlr.press/v162/janner22a.html.

Zheng Jiaoling, Xu Weifeng, Luo Qian, Dang Wanli, Geng Long, Gao Guokang, Ren Yulin, and Fan Xingyu. Decision preference alignment for large-scale agents based on reward model generation. In *ICLR 2025 Workshop on Bidirectional Human-AI Alignment*, 2025. URL https://openreview.net/forum?id=mQ1pLtdjbq.

Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d45e0bfb5a39477d56b55c0824200008-Abstract-Conference.html.

Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. *Trans. Mach. Learn. Res.*, 2025, 2025. URL https://openreview.net/forum?id=f7OkIurx4b.

Andreas Kontogiannis, Konstantinos Papathanasiou, Yi Shen, Giorgos Stamou, Michael M. Zavlanos, and George A. Vouros. Enhancing cooperative multi-agent reinforcement learning with state modelling and adversarial exploration. *CoRR*, abs/2505.05262, 2025. doi: 10.48550/ARXIV.2505.05262. URL https://doi.org/10.48550/arXiv.2505.05262.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5126–5137. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/373e4c5d8edfa8b74fd4b6791d0cf6dc-Paper.pdf.

Luc Le Mero, Dewei Yi, Mehrdad Dianati, and Alexandros Mouzakitis. A survey on imitation learning techniques for end-to-end autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14128–14147, 2022. doi: 10.1109/TITS.2022.3144867.

Pengyi Li, Jianye Hao, Hongyao Tang, Yan Zheng, and Xian Fu. RACE: improve multi-agent reinforcement learning with representation asymmetry and collaborative evolution. In Andreas

Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19490–19503. PMLR, 2023. URL https://proceedings.mlr.press/v202/li23i.html.

Lipowski Dorota Lipowska. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, 2012.

Iou-Jen Liu, Unnat Jain, Raymond A. Yeh, and Alexander G. Schwing. Cooperative exploration for multi-agent deep reinforcement learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6826–6836. PMLR, 2021. URL http://proceedings.mlr.press/v139/liu21j.html.

Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 22825–22855. PMLR, 2023. URL https://proceedings.mlr.press/v202/lu23d.html.

Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. MAVEN: multi-agent variational exploration. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 7611–7622, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/f816dc0acface7498e10496222e9db10-Abstract.html.

Paul Muller, Shayegan Omidshafiei, Mark Rowland, Karl Tuyls, Julien Pérolat, Siqi Liu, Daniel Hennes, Luke Marris, Marc Lanctot, Edward Hughes, Zhe Wang, Guy Lever, Nicolas Heess, Thore Graepel, and Rémi Munos. A generalized training approach for multiagent learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Bkl5kxrKDr.

Shayegan Omidshafiei, Christos Papadimitriou, Georgios Piliouras, Karl Tuyls, Mark Rowland, Jean-Baptiste Lespiau, Wojciech M Czarnecki, Marc Lanctot, Julien Perolat, and Remi Munos. α-rank: Multi-agent evaluation by evolution. *Scientific reports*, 9(1):9937, July 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-45619-9. URL https://europepmc.org/articles/PMC6617105.

Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and Rémi Munos. Count-based exploration with neural density models. *ArXiv*, abs/1703.01310, 2017. URL https://api.semanticscholar.org/CorpusID:2924063.

Pierre-Yves Oudeyer and Frédéric Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers Neurorobotics*, 1:6, 2007. doi: 10.3389/NEURO.12.006.2007. URL https://doi.org/10.3389/neuro.12.006.2007.

Matej Pecháč, Michal Chovanec, and Igor Farkaš. Self-supervised network distillation: An effective approach to exploration in sparse reward environments. *Neurocomputing*, 599:128033, 2024.

Sarah Perrin, Julien Pérolat, Mathieu Laurière, Matthieu Geist, Romuald Elie, and Olivier Pietquin. Fictitious play for mean field games: Continuous time analysis and applications. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/995ca733e3657ff9f5f3c823d73371e1-Abstract.html.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=mEpqHvbD2h.

David Silver, Satinder Singh, Doina Precup, and Richard S. Sutton. Reward is enough. *Artif. Intell.*, 299:103535, 2021. doi: 10.1016/J.ARTINT.2021.103535. URL https://doi.org/10.1016/j.artint.2021.103535.

Alexander L. Strehl and Michael L. Littman. An analysis of model-based interval estimation for markov decision processes. *J. Comput. Syst. Sci.*, 74(8):1309–1331, 2008. doi: 10.1016/J.JCSS.2007.08.009. URL https://doi.org/10.1016/j.jcss.2007.08.009.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. #exploration: A study of count-based exploration for deep reinforcement learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 2753–2762, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3a20f62a0af1aa152670bab3c602feed-Abstract.html.

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Joseph Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, L. Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander Sasha Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom Le Paine, Caglar Gulcehre, Ziyun Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy P. Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575:350 – 354, 2019. URL https://api.semanticscholar.org/CorpusID:204972004.

Tonghan Wang, Jianhao Wang, Yi Wu, and Chongjie Zhang. Influence-based multi-agent exploration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BJgy96EYvr.

Yongjie Wang, Yuchen Niu, Wenying Zhu, Wenqiang Chen, Qiong Li, and Tao Wang. Predicting pedestrian crossing behavior at unsignalized mid-block crosswalks using maximum entropy deep inverse reinforcement learning. *IEEE Trans. Intell. Transp. Syst.*, 25(5):3685–3698, 2024. doi: 10.1109/TITS.2023.3326276. URL https://doi.org/10.1109/TITS.2023.3326276.

Zhendong Wang, Jonathan J. Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

Wei Xiao, Tsun-Hsuan Wang, Chuang Gan, Ramin M. Hasani, Mathias Lechner, and Daniela Rus. Safediffuser: Safe planning with diffusion probabilistic models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=ig2wk7kK9J.

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. *Advances in neural information processing systems*, 35:24611–24624, 2022.

Guanwen Zeng, Jianxi Gao, Louis Shekhtman, Shengmin Guo, Weifeng Lv, Jianjun Wu, Hao Liu, Orr Levy, Daqing Li, Ziyou Gao, H. Eugene Stanley, and Shlomo Havlin. Multiple metastable network states in urban traffic. *Proceedings of the National Academy of Sciences*, 117(30):17528–17534, 2020. doi: 10.1073/pnas.1907493117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1907493117.

Lulu Zheng, Jiarui Chen, Jianhao Wang, Jiamin He, Yujing Hu, Yingfeng Chen, Changjie Fan, Yang Gao, and Chongjie Zhang. Episodic multi-agent reinforcement learning with curiosity-driven exploration. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 3757–3769, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/1e8ca836c962598551882e689265c1c5-Abstract.html.

Meng Zhou, Ziyu Liu, Pengwei Sui, Yixuan Li, and Yuk Ying Chung. Learning implicit credit assignment for cooperative multi-agent reinforcement learning. In *Neural Information Processing Systems*, 2020.

Mingyan Zhou, Biao Wang, and Xiatao Sun. Developing trajectory planning with behavioral cloning and proximal policy optimization for path-tracking and static obstacle nudging. *CoR-R*, abs/2409.05289, 2024. doi: 10.48550/ARXIV.2409.05289. URL https://doi.org/10.48550/arXiv.2409.05289.

Zhengbang Zhu, Hanye Zhao, Haoran He, Yichao Zhong, Shenyu Zhang, Yong Yu, and Weinan Zhang. Diffusion models for reinforcement learning: A survey. *CoRR*, abs/2311.01223, 2023. doi: 10.48550/ARXIV.2311.01223. URL https://doi.org/10.48550/arXiv.2311.01223.

Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. Madiff: Offline multi-agent learning with diffusion models. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/07e278a120830b10aae20cc600a8c07b-Abstract-Conference.html.

A Appendix

A.1 Implementation detail

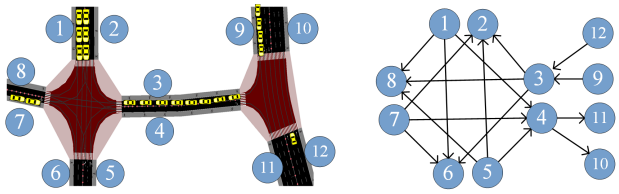
This subsection describes the specific Markov process used in the experiment. Before giving the Markov Decision Process, we first introduce the topology of road network.

Directed topology of a road network. The directed topology of a road network is defined as $G = (V, E, X)$. V represents the roads in the network, assume there are m roads in G. E denotes the connections between roads, assume there are $|E|$ connections. X indicates traffic status for each road.

Figure 6(a) illustrates the physical road network. Figure 6(b) shows its corresponding topological graph $G = (V, E, X)$. $V = \{road_1, \ldots, road_{12}\}$, $E = \{road_7 \rightarrow road_4, road_7 \rightarrow road_2, \ldots, road_4 \rightarrow road_{11}\}$. $road_7 \rightarrow road_4$ indicates vehicles can travel from $road_7$ to $road_4$. $X = \{queue_{road_1}, \ldots, queue_{road_{12}}\}$ represents the traffic status of roads, for example queue length of each road. Actually, G can be viewed as the complete MDP for learning $Q_\theta(x, a)$. We detail each component below.

State s. s represents the current state, indicating the road and queue length of the vehicle, $s = (road, queue_{road})$.

Action a. a is the navigation actions such as left turn, straight ahead, or right turn on the current road to the next road. For example, if the vehicle is in $road_4$, $a = (road_{10}, road_{11})$.



(a) Physical road network (b)Topological graph

Figure 6: Topological graph of road network.

Reward r**.** r is the reward obtained by steering to the next road at the current road.

Policy $Q_\theta(s,a)$**.** $Q_\theta(s,a)$ determines action a for each state s. $Q_\theta(s,a)$ maximizes cumulative discounted rewards, such that $Q_\theta(s,a) = \text{Argmax}\left[\sum_{t=1}^{T}\gamma^t r\left(s_t,a_t\right)\right]$. $\gamma \in [0,1)$ represents the discount factor.

For example, $((road_9, queue_{road_9}), road_3, (road_3, queue_{road_3}), r)$ is a piece of experience. It means that if vehicle in state $(road_9, queue_{road_9})$ takes action $road_3$, it will reach state $(road_3, queue_{road_3})$ and will get reward r. Let $\tau = \{(s_i, a_i, s_i')_t\}_{t=1}^{T}\}_{i=1}^{|E|}$ denote the offline trajectory for the vehicle. $|E|$ is the number of edges in G. Since the connection from road s to s' is fixed in a given traffic map, the number of experiences is exactly $|E|$.

However, if the local reward is not based on distance(such as "avoiding traffic lights" or "taking the favourite path"), reinforcement learning alone does not necessarily guarantee that the vehicle will reach the destination. This paper adopts a hybrid navigation policy of A* + Q-learning. For example, in Figure 6, when vehicle A travels from $road_9$ to $road_2$ using the A* alone. It selects the shortest path $road_9 \rightarrow road_3 \rightarrow road_2$. However, this path is the most congested. By combining A* with $Q_\theta(x,a)$, roads with longer queue lengths are assigned with lower rewards. The optimal path changes to a longer but less congested route: $road_9 \rightarrow road_1 1 \rightarrow ... \rightarrow road_5 \rightarrow road_2$.

A* + Q-learning navigation policy. The traditional A* navigation policy defines the total cost function as $r(s) = g(s) + h(s)$. $g(s)$ represents the actual cost from the starting point to the current node. $h(s)$ is the heuristic estimation function that estimates the potential cost to the endpoint. We designed $h(s)$ in Formula 12 by using an A* + Q-learning hybrid mechanism that not only meets routing requirements but also reflecting the vehicle's unique preferences.

$$h(s) = \|loc(s_{\text{current}}) - loc(s_{\text{end}})\|_2 - \lambda \cdot Q_\theta(s,a) \tag{12}$$

where $\|loc(s_{\text{current}}) - loc(s_{\text{end}})\|_2$ is the Manhattan distance between the current road of the vehicle and the destination of the vehicle. λ is an adjustable hyper-parameter. $Q_\theta(s,a)$ is the preference policy learned by reinforcement learning. s is the current state of the vehicle. a is the navigation actions such as turning left to the next road. A* tries to minimize the cost of the path, while DQN tries to maximize the reward. So, we use the negative values of $Q_\theta(s,a)$ in Formula 12.

We employ offline Deep Q-learning (Kostrikov et al., 2022) to train $Q_\theta(x,a)$. The training process is not performed in real-time but completed once the vehicle first enters the road network by using the offline dataset.

A.2 Experimental results in scenario 2 and scenario 3

Figure 7∼9 shows the results. The horizontal axis represents the minutes. The vertical axis represents the MQL value of a specific road.

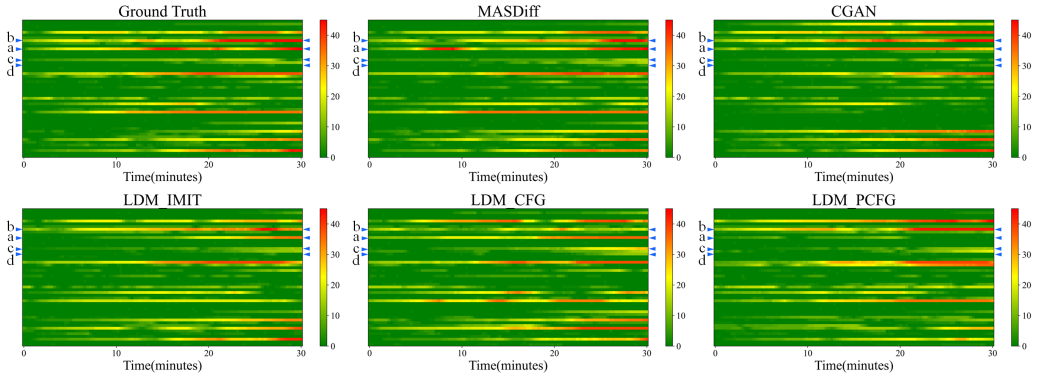


Figure 7: $\{MQL_t^*\}_{t=1}^{30}$ and $\{MQL_t\}_{t=1}^{30}$ of selected roads in scenario 1.

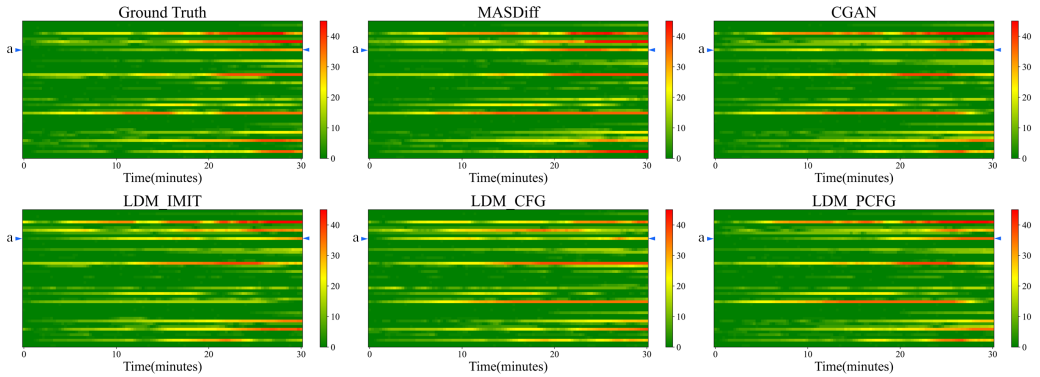


Figure 8: $\{MQL_t^*\}_{t=1}^{30}$ and $\{MQL_t\}_{t=1}^{30}$ of selected roads in scenario 2.

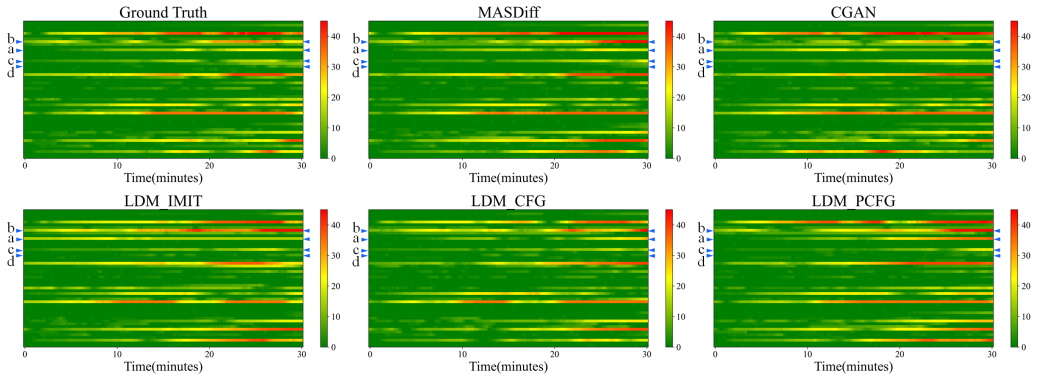


Figure 9: $\{MQL_t^*\}_{t=1}^{30}$ and $\{MQL_t\}_{t=1}^{30}$ of selected roads in scenario 3.