A Hybrid Framework of Reinforcement Learning and Physics-Informed Deep Learning for Spatiotemporal Mean Field Games

Xu Chen
Columbia University
New York, United States
xc2412@columbia.edu

Shuo Liu
Columbia University
New York, United States
sl4921@columbia.edu

Xuan Di
Columbia University
New York, United States
sharon.di@columbia.edu

ABSTRACT

Mean field games (MFG) are developed to solve equilibria in multi-agent systems (MAS) with many agents. The majority of literature on MFGs is focused on finite states and actions. In many engineering applications such as autonomous driving, however, each agent (e.g., an autonomous vehicle) makes a continuous-time-space (or spatiotemporal dynamic) decision to optimize a nonlinear cumulative reward. In this paper, we focus on a class of generic MFGs with continuous states and actions defined over a spatiotemporal domain for a finite horizon, named "spatiotemporal MFG (ST-MFG)." The mean field equilibria (MFE) for such games are challenging to solve using numerical methods to meet a satisfactory resolution in time and space, while it is critical to deploy smooth dynamic control in autonomous driving. Thus, we propose two methods, one is a joint reinforcement learning (RL) and machine learning framework, which iteratively solves agents' optimal policies using RL, and propagates population density using physics-informed deep learning (PIDL). The other is a pure PIDL framework that updates agents' states and population density altogether using deep neural networks. Both the proposed methods are mesh-free (i.e., not restricted by mesh granularity), and have shown to be efficient in learning equilibria in autonomous driving MFGs. The PIDL method alone is faster to train than the RL-PIDL integrated method, when the environment dynamic is known.

KEYWORDS

Reinforcement Learning; Physics-Informed Deep Learning; Mean Field Games

ACM Reference Format:
Xu Chen, Shuo Liu, and Xuan Di. 2023. A Hybrid Framework of Reinforcement Learning and Physics- Informed Deep Learning for Spatiotemporal Mean Field Games. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

With a large number of interacting agents in a multi-agent system (MAS), agents' decision-making processes could be computationally intractable. Mean field games (MFGs) are developed to solve agents' dynamic decision-making behaviors with conflicting goals, using a population distribution to represent the state of many individual agents [9, 10, 26, 29]. At mean field equilibria (MFE), an agent's optimal strategy coincides with the population density. MFGs have been widely studied in engineering, economics, and finance since its inception. Readers can refer to [2] for more details.

In this paper, we focus on a class of generic MFGs with continuous state and action spaces defined over a spatiotemporal domain for a finite horizon, named "spatiotemporal MFG (ST-MFG)." It models the continuous-time decision making of agents and their interactions across a continuous space over a finite horizon. It belongs to non-stationary mean field games where optimal policies of agents evolve with time. This is motivated by engineering and robotics applications such as autonomous driving [22, 24, 25], in which agents (e.g., autonomous vehicles) make dynamic decisions in time and space to optimize a nonlinear and possibly non-separable (i.e., a cross term between agents' control and the mass density in the cost functional) cumulative reward. The mean field equilibria (MFE) for such games are challenging to solve using numerical methods due to its infinite number of states and actions. In order to solve the spatiotemporal (ST) dynamics of population state and agents' decision-making, we adopt a hybrid framework, i.e., reinforcement learning (RL) coupled with physics-informed deep learning (PIDL), which combines both model-driven and data-driven neural networks.

Assuming agents are anonymous, mean field approximation can be applied to exploit the "smoothing" effect of large numbers of interacting individuals. At equilibrium, each player interacts and reacts only to a "mass" which results from the aggregate effect of all the players nearby. The MFG is thus a micro-macro model that allows one to define individuals on a microscopic level as rational, utility-optimizing agents while translating their rich microscopic behaviors to a macroscopic scale. It consists of two coupled partial differential equations (PDEs):

(1) *Agent dynamic*: individuals' dynamics using optimal control, i.e, a backward Hamilton-Jacobi-Bellman (HJB) equation;
(2) *Mass dynamic*: system evolution arising from each individual's choices, i.e, a forward Fokker-Planck-Komogorov (FPK) equation.

These two coupling equations characterize the evolution of the system's dynamics. At MFE, an agent's optimal strategy coincides with the population density.

MFE is challenging to solve due to its forward-backward structure. The existing literature primarily employs three types of numerical methods, namely, fixed-point iteration [13, 15, 48], variational method [6, 14, 28], and Newton's method [1, 3]. The former two require special structures of MFGs, which do not directly apply to ST-MFGs [24]. While the Newton's method does not impose requirements on the length of planning horizon nor the cost function, it may fail to converge if one does not have a good initial guess to the solution. So tricks such as a multigrid preconditioned algorithm [24] are needed to improve the convergence. All the

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

aforementioned numerical methods require the spatial-temporal discretization of a dynamic system, and accordingly, the mesh size of the discretized system could influence computational efficiency and accuracy. Thus, these methods could suffer from the complexity and dimension of state and action spaces [31].

To tackle the above challenges, we resort to learning based methods to solve MFE for its mesh-free scheme and efficiency in handling interactions among agents in complex environments.

The main contributions of this paper include:

- Propose two methods to solve time-dependent non-stationary control policies with continuous states and actions in MFGs: a joint framework of RL and PIDL and a pure PIDL framework; We establish the linkage between two methods with known dynamics in the MFG system.
- Develop two algorithms [MFG-RL-PIDL] and [MFG-pure-PIDL] for proposed frameworks to find MFE in ST-MFGs; [MFG-RL-PIDL] unifies the training of a physics-informed neural networks (PINN), actor and policy networks in the RL module; [MFG-pure-PIDL] replaces the RL module with a PIDL module to speed up the training.
- Validate developed algorithms in autonomous driving games with different cost functional forms, including Monotone MFGs and Non-monotone MFGs.

The rest of this paper is organized as follows: Section 2 presents related work and preliminaries about ST-MFG. Section 3 proposes a RL-PIDL framework for ST-MFGs. Section 4 proposes a pure PIDL framework. Section 5 discusses the linkage between proposed methods. Section 6 demonstrates numerical experiments conducted on autonomous driving games. Section 7 concludes.

2 BACKGROUND

2.1 Related Work

There is a growing trend of applying RL methods to find equilibria in MFGs [27, 46, 47]. To accommodate continuous population states and agent actions, deep deterministic policy gradient (DDPG) [16], normalizing Flow (NF) [36], actor and critic (A2C) [32, 42] are adopted. To stabilize the agent's policy learning, fictitious play (FP) is introduced into the learning framework for MFGs by incorporating empirical best responses during the learning process into the decision making [11, 31, 35–37, 44]. Other methods to stabilize agents' behavior include regularization [4, 45], policy evaluation [20, 34], and population-based training [33].

Deep learning (DL) methods have also been applied to MFGs [12, 18] with neural networks to approximate system dynamics in a mesh-free scheme [38]. Various neural architectures have been leveraged to solve high-dimensional PDE problems [40, 43].

The majority of aforementioned studies that used machine learning methods to solve MFGs, however, relies on stringent assumptions such as stationarity [19, 41], discrete actions or states [5, 7, 8, 21], as well as reward monotonicity [17, 37].

2.2 Spatiotemporal MFG (ST-MFG)

Spatiotemporal MFG (ST-MFG) refers to a class of MFGs with both the population state and agents' actions defined in a spatiotemporal domain over a finite horizon. The reward or cost arising from agents' actions negatively depend on the population density, indicating a *congestion effect*. ST-MFG is non-stationary because optimal policies of agents evolve with time.

Definition 2.1. ***ST-MFG***
Define a finite planning horizon $\mathcal{T} = [0, T]$ where $T \in [0, \infty)$. A total of N agents, indexed by $n = \{1, 2, \cdots, N\}$, are moving in a 1- or 2-dimension space, denoted by $\mathcal{X}$. Their positions at time t are denoted as $\mathbf{x}(t) = [x_1(t), x_2(t), \cdots, x_N(t)]$. Agent $n \in N$ controls $u_n(t) \in \mathcal{A}$ where $\mathcal{A}$ is the feasible action set to minimize its cost functional: $\forall n = 1, \cdots, N$,

$$J_n^N(u_n, u_{-n}) = \int_0^T \underbrace{f_n^N(u_n(t), x_n(t), x_{-n}(t))}_{\text{cost function}} dt + \underbrace{V_T(x_n(T))}_{\text{terminal cost}}. \tag{1}$$

A Nash equilibrium of the N-player mean field type differential game is a tuple of controls $u_1^*(t), u_2^*(t), \ldots, u_N^*(t)$ satisfying

$$J_n^N(u_n^*, u_{-n}^*) \leq J_n^N(u_n, u_{-n}^*), \forall n = 1, \cdots, N. \tag{2}$$

As $N \to \infty$, the optimal cost of a generic agent from x at time t becomes:

$$V(x,t) = min_u\{\int_t^T f(u(x(\tau),\tau), \rho(x(\tau),\tau))d\tau + V(x(T),T)\} \tag{3}$$

where, $u(x(\tau), \tau)$ is the control of a generic agent. The agent state $x(\tau), \forall \tau \in \mathcal{T}$ is updated based on the agent dynamics $\dot{x}(\tau) = u(x(\tau), \tau)$. $x(\tau)$ is the agent position by time τ and we denote $x = x(\tau), x \in \mathcal{X}$. $\rho(x,t), \forall (x,t) \in \mathcal{X} \times \mathcal{T}$ is the population density of all agents in the system (i.e., mean-field state). $f(u, \rho)$ is the cost function. $V(x,t), \forall (x,t) \in \mathcal{X} \times \mathcal{T}$ is the value function for each individual agent, which can be interpreted as the minimum cost of an agent when starting from position x by time t. $V(x(T), T)$ denotes the terminal cost. We have $V(x,T) = \hat{V}(x), \forall x \in \mathcal{X}$.

We denote partial derivatives of $\rho(x,t)$ with respect to x, t as ρ_x and ρ_t, respectively. It is the same for V and u. The population dynamics can be captured by a Fokker-Planck equation (FPK):

$$\text{(FPK)}\ \rho_t + (\rho \cdot u)_x = 0, \tag{4}$$

which describes the evolution of population density $\rho(x,t)$ according to the control $u(x,t)$ of agents. The population density starts from initial density $\rho(x,0) \equiv \hat{\rho}(x), \forall x \in \mathcal{X}$. Equation 3 can be reformulated as a Hamilton-Jacobi equation:

$$\text{(HJB)}\ V_t + \min_u\{f(u,\rho) + uV_x\} = 0, \tag{5}$$

which captures the relationship between the cost $V(x,t)$ and the agent's control $u(x,t)$.

We reformulate the MFG system as:

$$\begin{aligned}
&[\text{ST-MFG}]: \\
&\min_u \int_t^T f(u(x(\tau),\tau), \rho(x(\tau),\tau))d\tau + V(x(T),T), \forall (x,t) \in \mathcal{X} \times \mathcal{T} \\
&s.t.\ \dot{x(t)} = u(x(t),t), x(t) \equiv x, \forall t \in \mathcal{T},\ \text{(agent dynamics)} \\
&\quad \rho_t + (\rho \cdot u)_x = 0, \forall (x,t) \in \mathcal{X} \times \mathcal{T},\ \text{(population dynamics)} \\
&\quad \rho(x,0) \equiv \hat{\rho}(x), \forall x \in \mathcal{X},\ \text{(initial density)} \\
&\quad V(x,T) \equiv \hat{V}(x), \forall x \in \mathcal{X}.\ \text{(terminal cost)}
\end{aligned} \tag{6}$$

Denote the equilibrium solution by $\rho^*(x,t)$ and $u^*(x,t)$. The optimal velocity field $u^*(x,t)$ is our primary focus and will thus be referred as the mean field equilibrium (MFE) in the subsequent analysis.

The ST-MFG can be categorized based on the following critera:

(1) *Non-stationarity*: Note that ST-MFG is non-stationary, or "evolutive MFG" [30]. The policy of the representative agent and the mean field state evolve as time progresses.
(2) *Finite time horizon*: We study ST-MFG with finite time horizon. It is challenging to solve ST-MFG with infinite time horizon. This is because for non-stationary MFGs with infinite time horizon, the value function $V(x,\infty)$ could go to infinity [29]. We leave this for future work.
(3) *First-order MFG*: Since agents like autonomous cars or robots may not appear or disappear randomly in a conserved system, there is no stochasticity in the FPK equation. Thus, The FPK equation is reduced to a first-order deterministic continuity/transport equation.

2.3 Solution Concepts

Definition 2.2. ***Mean Field Equilibrium (MFE)***
In ST-MFG, $(u^*(x,t),\rho^*(x,t)),\forall(x,t)\in\mathcal{X}\times\mathcal{T}$ is called an MFE if following conditions hold:

$$\begin{cases} (FPK) \quad \rho^*_t + (\rho^*\cdot u^*)_x = 0 & \text{(7a)}\\ \text{(HJB)} \quad V^*_t + u^*V^*_x + f(u^*,\rho^*) = 0 & \text{(7b)}\\ \qquad\quad u^* = g^*_p(V^*_x,\rho^*) & \text{(7c)}\end{cases}$$

where $g^*_p(V_x,\rho) = argmin_p\{f(p,\rho)+pV_x\}, V_x\in\mathbb{R}$. For simplicity, we omit discussion on solution properties. Readers can refer to [23, 24] for more details.

Definition 2.3. ***Monotone MFG***
An MFG is called a monotone MFG [37] is following conditions holds:

$$\begin{cases} (Separable) \quad f(u,\rho) = \tilde{f}(u) + \bar{f}(\rho) & \text{(8a)}\\ \text{(Monotone)} \quad \forall\rho,\rho',(\rho-\rho')\cdot(\bar{f}(\rho)-\bar{f}(\rho'))\geqslant 0 & \text{(8b)}\end{cases}$$

Equation 8a indicates that the cost function $f(u,\rho)$ has a separable structure. There is no cross product between u and ρ.

In this paper, we also consider another structure of the cost function: non-separable cost. We introduce a cross product into the cost functional between the agent action u and the population density ρ to reflect the congestion effect, demonstrating the punishment to the agents who select the same actions or end up in close proximity under a policy. The more the agents stay in the same neighborhood, the more congested that area is. This also renders the ST-MFG not as a potential game, and thus, existing ML methods to solve potential MFGs do not apply. We investigate ST-MFGs with three cost functions in numerical results (Section 6) where one is a Monotone MFG and the remaining two are not Monotone MFGs.

2.4 Numerical Method

To solve ST-MFG, [24] discretized the spatiotemporal domain $\mathcal{X}\times\mathcal{T}$ by solution granularity Δx and Δt according to the Courant Friedrichs Lewy (CFL) condition where $u_{max}\cdot\Delta t\leqslant\Delta x$, and then solved a system of equations in MATLAB. However, the numerical method encounters several issues: First, it cannot meet a satisfactory resolution in time and space. A small spatiotemporal granularity Δx and Δt would significantly increase the problem scale, making the ST-MFG not solvable. The numerical method with satisfactory resolution only works in small-size domains. Second, the structure of cost functions may impact the performance of numerical methods to find game equilibria. To tackle these challenges, this paper leverages learning frameworks to solve ST-MFG.

3 RL-PIDL FRAMEWORK OVERVIEW

In this section, we propose a joint framework of RL and PIDL to learn ST-MFG. In this framework, the evolution of population density (i.e., mean-field state) is approximated by physics-informed neural networks (PINN) while the decision making of the generic agent is captured by a single-agent RL module.

Figure 1 demonstrates the working flow of the RL-PIDL method: Three neural networks ρ-Net, u-Net and V-Net are utilized to represent the population density, the agent's control and cost, respectively. u-Net and V-Net are actor and critic networks in a single-agent RL module given the population distribution ρ and agent dynamics in the environment. ρ-Net is approximated by a PIDL module given the physical rule (FPK) regarding the relationship between the evolution of population and the agent control. The RL and PIDL modules internally depend on each other. The policy learning of the generic agent triggers the update of system dynamics, which in turn influences the learning process of the agent. A fictitious play module is adopted to stabilize policy learning. We now introduce RL and PIDL modules separately.

3.1 RL for Agent Optimal Control

Given the population density $\rho(x,t),\forall(x,t)\in\mathcal{X}\times\mathcal{T}$ (mean-field state), the dynamic control problem of the generic agent can be formulated into the following RL scheme.

(1) **State** $s\in\mathcal{S}$: The state of the representative agent $s\equiv(x,t)$ indicates that at time t, the agent arrives at position x.
(2) **Action** $u\in\mathcal{U}$: $u(x,t)$ is the control of the agent at position x by time t. $\mathcal{U}\equiv[u_{min},u_{max}]$. In this paper, we assume the agent adopts a deterministic policy. The u-Net is a deterministic policy network (Figure 1), parameterized by ω. In this work, we apply Deep Deterministic Policy Gradient (DDPG) method to the RL module.
(3) **Transition** $s\rightarrow s'$: The agent's action triggers the state transition $(x,t)\rightarrow(x',t')$, where $x'=x+u(x,t)\cdot\delta t$, $t'=t+\delta t$. δt is the time interval in the decision making process of the agent.
(4) **Reward** r: The reward is the congestion cost incurred by interaction with population in the system, i.e., $r(u,\rho) = f(u(x,t),\rho(x,t))\delta t$.
(5) **Value function** V: The value function $V(x,t)$ represents the total cost of the agent starting from location x by time t. $V(x,t)$ is captured the critic network V-Net, parameterized by η. Mathematically,

$$V^*(x,t) = \min_u[r(u(x,t),\rho(x,t)) + V^*(x',t')] \tag{9}$$

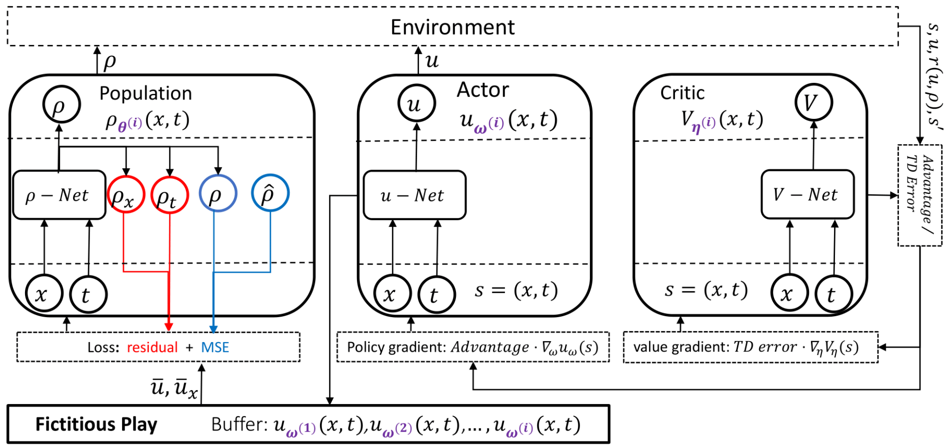


Figure 1: RL-PIDL Framework

where V^* is the minimum cost of the agent.

3.2 PIDL for Population Density Propagation

We now introduce physics-informed deep learning (PIDL) to approximate population dynamics in the MFG system. The PIDL module adopts a hybrid deep learning framework, which combines both model-driven and data-driven neural networks [39]. The neural network architecture in PIDL to capture system dynamics is illustrated in Figure 1. ρ-Net is parameterized by θ. The input is (x, t) and output is the population density at location x by time t.

The training of ρ-Net is guided by two parts in the loss function: residual and the mean square errors (MSE). The residual (marked in red) leverages physical rules that exist in the dynamic system while MSE (marked in blue) is obtained by the gap between PINN and observed data points.

In ST-MFG, the evolution of population state given the control of agents $u(x, t)$ follows the FPK equation (4), which indicates the physical rule regarding the spatiotemporal distribution of population. We use the physical rule to guide the training of $\rho_\theta(x, t)$ by the following residual:

$$q_\theta(x,t) = \frac{\partial \rho_\theta(x,t)}{\partial t} + \frac{\partial[\rho_\theta(x,t)\bar{u}(x,t)]}{\partial x} \tag{10}$$

The PINN calculates the residual $q_\theta(x, t)$. When $\rho_\theta(x, t)$ becomes close to $\rho(x, t)$ satisfying the FPK equation, the residual gets close to zero. $\bar{u}(x, t)$ represents the average policy of the agent obtained by historical policies in the fictitious play module.

The observed data in the PIDL framework comes from the initial distribution of population $\rho(x, 0) \equiv \hat{\rho}(x), \forall x \in \mathcal{X}$ (marked in blue circles in Figure 1). The training of $\rho_\theta(x, t)$ based on observed data follows the traditional supervised learning scheme. The mean square errors (MSEs) are:

$$\text{MSE}_o = \frac{1}{N_o}\sum_{k=1}^{N_o}(\rho_\theta(x_o^k, 0) - \hat{\rho}(x_o^k))^2, x_o^k \in \mathcal{X} \tag{11}$$

where N_o is the size of data sample. $\hat{\rho}(x_o^k, 0)$ is the observed population density at position x_o^k by time 0.

The loss function used in the PIDL module is computed as:

$$\text{Loss} = \beta_o \text{MSE}_o + \beta_c r_c \tag{12}$$

where β_o and β_c are hyperparameters, representing the weight of MSE and residual in the loss function, respectively. $\beta_o\text{MSE}_o$ measures the data discrepancy and $\beta_c r_c$ denotes the physical discrepancy in the training of PIDL. Note that the observed data can only be sampled from the state space when $t = 0$ given the fact that the initial condition of the FPK equation is known.

3.3 Fictitious Play for Policy Stabilization

Fictitious play (FP) is utilized to stabilize policy learning. We add an FP buffer between the policy and population network. The FP buffer is used to store all historical policies from the actor. The residual (Equation 10) for the training of the population network is calculated based on the average policy from the FP buffer. Mathematically,

$$\bar{u}(x,t) = \sum_{j=1}^{i} u_{\omega^{(j)}}(x,t), \forall (x,t) \in \mathcal{X} \times \mathcal{T} \tag{13}$$

where $u_{\omega^{(j)}}(x, t)$ is the agent policy (i.e., u-Net) at the jth iteration during the training process.

3.4 Learning Algorithm

In this subsection, we develop a learning algorithm where the training of ρ-Net in the PIDL module is coupled with u-Net and V-Net in the RL module. We first discuss the solution granularity in the learning scheme.

Solution granularity

We use finite difference based on the CFL condition (i.e., $u_{max}\Delta t \leqslant \Delta x$) instead of autograd mechanism, to denote the partial derivative information of neural networks and calculate residuals for PIDL.

Therefore, equation 10 can be formulated as

$$q_\theta(x,t) = \frac{\rho_\theta(x,t+\phi\Delta t) - \rho_\theta(x,t)}{\phi\Delta t} \text{ (i.e., } \rho_t)$$
$$+ \frac{\rho_\theta(x,t)u(x,t) - \rho_\theta(x-\phi\Delta x,t)\bar{u}(x-\phi\Delta x,t)}{\phi\Delta x} \text{ (i.e., } (\rho u)_x) \tag{14}$$

where Δx and Δt represent the spatiotemporal granularity. In this work, we assume $u_{max} = 1$ and $\Delta x = \Delta t$. $\phi \in [1, \bar{\phi}]$ is a random step size when calculating partial derivatives in the residual, which can help stabilize the training of the PINN without sampling too many states $s = (x,t)$ from state space.

Algorithm 1 MFG-RL-PIDL

Initialization: Population network ρ-Net: $\rho_{\theta^{(0)}}(s)$; Actor network u-Net: $u_{\omega^{(0)}}(s)$ and critic network V-Net: $V_{\eta^{(0)}}(s)$.
for $i \leftarrow 0$ to I **do**
  Sample a batch of states $\mathbf{s}$ from state space $\mathcal{X} \times \mathcal{T}$;
  for each state s_l in $\mathbf{s}$ **do** *—RL - the representative agent*
    Select u according $u_{\omega^{(i)}}(s_l)$;
    Obtain ρ according $\rho_{\theta^{(i)}}(s_l)$;
    Execute u and observe reward $r(u,\rho)$;
    Update state $s_l \to s'_l$;
    Obtain value function: $V_{\eta^{(i)}}(s), V_{\eta^{(i)}}(s')$.
  end for
  Calculate the advantage (Equation 15);
  Store the actor network $u_{\omega^{(i)}}(s)$ into buffer. *—FP*
  Compute $\bar{u}$ (Equation 13);
  Obtain MSE_o (Equation 11); *—PIDL - Population*
  Obtain residual (Equation 14 and 16);
  Update ρ-Net, u-Net and V-Net and obtain $\rho_{\theta^{(i+1)}}(s)$, $u_{\omega^{(i+1)}}(s)$ and $V_{\eta^{(i+1)}}(s)$;
  Check convergence (Equation 17).
end for
Output u, ρ

We now look into the proposed learning algorithm [MFG-RL-PIDL], which is summarized in Algorithm (1). We first initialize ρ-Net, u-Net and V-Net, parameterized by $\theta^{(0)}$, $\omega^{(0)}$ and $\eta^{(0)}$, respectively. During the ith iteration of the training process, we first sample a batch of states $\mathbf{s}$ from state space $\mathcal{X} \times \mathcal{T}$. For simplicity, we assume agents are moving in a 1-dimension space $\mathcal{X} = [0, X]$. We divide $\mathcal{X}$ and $\mathcal{T}$ into n same pieces:

$$0 = x_0 < x_1 < \cdots < x_n = X,$$
$$0 = t_0 < t_1 < \cdots < t_n = T,$$

A batch of states $\mathbf{s}$ with size $n \times n$ is constructed as follows: $\forall l, k = 1, 2, ..., n$, (x^l, t^k) is sampled from $[x_{l-1}, x_l] \times [t_{k-1}, t_k]$ and we assume x^l and t^k are uniformly distributed on $[x_{l-1}, x_l]$ and $[t_{k-1}, t_k]$. For each state s_l in the batch, the agent's action generated by u-Net triggers the state transition $s_l \to s'_l$. Accordingly, the advantage in the RL module is calculated as:

$$\frac{1}{\mathcal{K}(\mathbf{s})} \sum_{l=1}^{\mathcal{K}(\mathbf{s})} [r(u(s_l), \rho(s_l)) + V_\eta(s'_l) - V_\eta(s_l)] \tag{15}$$

where $\mathcal{K}(\mathbf{s})$ is the batch size and $s_l \in \mathbf{s}$. $s'_l, l = 1, ..., \mathcal{K}(\mathbf{s})$ is the new state after the agent selects her action at state s_l. A fictitious play buffer is utilized to store historical policy networks. We calculate the average policy based on the fictitious play buffer and obtain the MSE and residual in the loss function. The residual of ρ-net in the PIDL module based on the batch of states is calculated as:

$$\frac{1}{\mathcal{K}(\mathbf{s})} \sum_{l=1}^{\mathcal{K}(\mathbf{s})} q_\theta(s_l) \tag{16}$$

We then update ρ-Net, u-Net and V-Net according to loss function, policy and value gradient, respectively. We check the following convergence conditions for population-agent pair (ρ, u):

$$\frac{1}{N} \left| \sum_{k=1}^{N} u_{\omega^{(i)}}(x^k, t^k) - \sum_{k=1}^{N} u_{\omega^{(i)}}(x^k, t^k) \right| < \epsilon_u$$
$$\frac{1}{N} \left| \sum_{k=1}^{N} \rho_{\theta^{(i)}}(x^k, t^k) - \sum_{k=1}^{N} \rho_{\theta^{(i-1)}}(x^k, t^k) \right| < \epsilon_\rho \tag{17}$$

The training process moves on to the next iteration till the convergence conditions hold. The algorithm is implemented in PyTorch.

4 PURE PIDL FRAMEWORK OVERVIEW

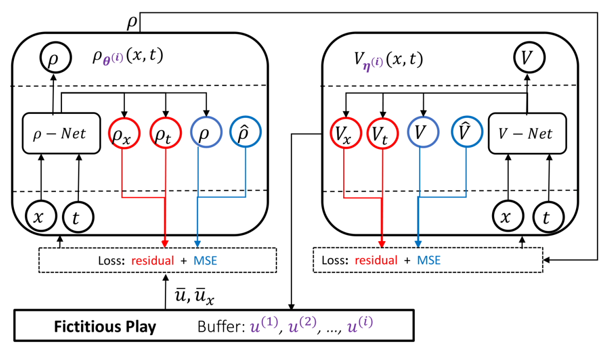


Figure 2: Pure PIDL Framework

In this section, we propose another learning framework by leveraging the PIDL method alone. This framework adopts two PINNs: ρ-Net and V-Net for FPK and HJB equations, respectively. Figure 2 demonstrates the working flow of the pure PIDL framework. The left ρ-Net approximates the population propagation and the right V-Net approximates the cost of the generic agent given the population distribution and agent control over the environment.

4.1 PIDL for Population Density Propagation

In the pure PIDL framework, ρ-Net works as same as the PIDL module in Section 3.2. We omit discussion for simplicity.

4.2 PIDL for Agent Optimal Control

In ST-MFG, the cost of the generic agent follows the HJB equation (7b and 7c). We use the physical rule to guide the training of $V_\eta(x,t)$ by the following residual:

$$q_\eta(x,t) = \frac{\partial V_\eta(x,t)}{\partial t} + u\frac{\partial V_\eta(x,t)}{\partial x} + f(u,\rho) \tag{18}$$

When $V_\eta(x,t)$ becomes close to $V(x,t)$ satisfying the HJB equation, the residual gets close to zero. The observed data in the PIDL framework comes from the condition about terminal cost $V(x,T) \equiv \hat{V}(x), \forall x \in \mathcal{X}$ (marked in blue circles in Figure 2). The mean square errors (MSEs) are:

$$\mathrm{MSE}_o = \frac{1}{N_o}\sum_{k=1}^{N_o}(V_\eta(x_o^k,T) - \hat{V}(x_o^k))^2, x_o^k \in \mathcal{X} \tag{19}$$

The loss function used to train the V-Net consists of the MSE and residual defined in Equation 18 and 19.

Remark. According to Equation 7c ($u = argmin_p\{f(p,\rho) + pV_x\}$), the agent control u can be directly obtained by the cost function and the partial derivative V_x of V-Net. We store $u^{(i)}$ at the ith iteration into the fictitious play module during the training process.

4.3 Learning Algorithm

We briefly introduce the learning algorithm for the pure PIDL framework, which is summarized in Algorithm (2). We first initialize ρ-Net and V-Net, parameterized by $\theta^{(0)}$ and $\eta^{(0)}$, respectively. During the ith iteration of the training process, we first sample a batch of states $\mathbf{s}$ from state space $\mathcal{X}\times\mathcal{T}$. The residual and MSE for $V-net$ are calculated according to Equation 18 and 19, respectively. We calculate the average policy based on the fictitious play buffer and then obtain the residual and MSE for ρ-Net. ρ-Net and V-Net are updated according to their loss functions. We check the convergence according to Equation 17.

Algorithm 2 MFG-Pure-PIDL

```
Initialization: ρ-Net: ρ_θ(0)(s) and V-Net: V_η(0)(s).
for i ← 0 to I do
    Sample a batch of state s from X × T;
    Obtain MSE_o (Equ 19) and residual (Equ 18); — PIDL-V-Net
    Calculate u^(i) (Equ 7c);
    Store u^(i) into buffer and compute ū;        — FP
    Obtain MSE_o and residual for ρ-Net           — PIDL-ρ-Net
    Update ρ-Net and V-Net according to loss function and
    obtain ρ_θ(i+1)(s) and V_η(i+1)(s);
    Check convergence.
end for
Output u, ρ
```

5 LINKAGE BETWEEN TWO METHODS

The difference between two proposed frameworks lies in how to denote the HJB equation in the MFG system. The RL-PIDL module leverages an agent-based learning scheme to study the optimal control problem while the PIDL module adopts a PINN to approximate the HJB equation. In this section, we investigate the linkage between these two methods.

Proposition 5.1. If the spatiotemporal granularity satisfies CFL condition (i.e., $u_{max}\Delta t \leqslant \Delta x$), the residual $V_t + uV_x + f(u,\rho) = 0$ of the PINN is equivalent to $r + V(s') - V(s) = 0$ where $r + V(s') - V(s)$ is the advantage for the critic network in the RL module.

Proof.

$$\begin{aligned} & r + V(s') - V(s) = 0 \\ \to & V(x,t) = f(u,\rho)\Delta t + V(x',t') \\ \to & V(x,t) = f(u,\rho)\Delta t \\ & \underbrace{+ V(x,t+\Delta t) + u\frac{\Delta t}{\Delta x}[V(x+\Delta x, t+\Delta t) - V(x,t+\Delta t)]}_{\text{Approximate } V(x',t') \text{ by linear interpolation [24]}} \end{aligned}$$

$0 \leqslant u\frac{\Delta t}{\Delta x} \leqslant 1$ holds because $u\frac{\Delta t}{\Delta x} \leqslant u_{max}\frac{\Delta t}{\Delta x} \leqslant 1$ (CFL condition). We then have

$$\frac{V(x,t) - V(x,t+\Delta t)}{\Delta t} = f(u,\rho) + u\frac{V(x+\Delta x,t+\Delta t) - V(x,t+\Delta t)}{\Delta x}$$

When $\Delta t, \Delta x \to 0$, $-V_t = f(u,\rho) + uV_x$. Therefore, Proposition 5.1 holds. □

Remark.
(1) Proposition 5.1 shows that the critic network in the RL module works as same as the V-net in the PIDL module. It means that the RL module captures the physical rule regarding the relationship between the agent control and total cost.
(2) The RL-PIDL framework can be replaced by the pure PIDL framework if the dynamics are known. With Equation 7c, the control of the generic agent can be directly obtained by V-Net in the PIDL module without utilizing a policy network, which speeds up the training process.

6 NUMERICAL EXPERIMENTS

In this section, we apply proposed methods to autonomous driving system. We first introduce an ST-MFG regarding the speed control of autonomous vehicles (AVs) and implement algorithms [MFG-RL-PIDL] and [MFG-Pure-PIDL] on the speed control problem with different cost structures. We then make a comparison of the numerical method (Section 2.4) and our methods.

6.1 Problem Statement

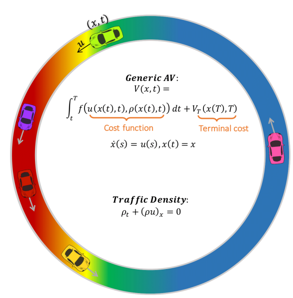


Figure 3: Speed control for AVs

In Fogure 3, we consider a generic AV staring from position x at time t. The vehicle's speed control is denoted by u and the goal of the generic AV is to minimize total travel cost $V(x,t), \forall(x,t) \in \mathcal{X}\times\mathcal{T}$ by selecting optimal speed. The decision making of the generic AV follows the HJB equation (5). When all cars follow the optimal speed control, the aggregated density distribution $\rho(x,t)$ (i.e., population state) evolves. Density ρ follows the FPK equation (4), which is also called continuity equation [24], demonstrating the road density in traffic flow models. In this work, we adopts a ring road with length 1 (i.e., $\mathcal{X} = [0, 1]$) as the traffic environment. It means positions $x = 0$ and $x = 1$ are the same. Vehicles are allowed to keep moving along the ring road until time T. The ring road scenario can be easily extended to any road segments/links. We implement our methods on the ST-MFG with three cost functions:

(1) ***LWR***: The Lighthill-Whitham-Richards (LWR) model is a traditional traffic flow model where the driving objective is to maintain some desired speed. The cost function is:

$$f(u, \rho) = \frac{1}{2}(U(\rho) - u)^2 \tag{20}$$

where $U(\rho)$ is an arbitrary desired speed function with respect to density ρ. It is straightforward to find that the analytical solution of the LWR model is $u = U(\rho)$, which means at MFE, vehicles maintain the desired speed on roads. We denote the ST-MFG as [ST-MFG-LWR]

(2) ***Separable***: The separable cost function can be written as the sum of two univariate functions with respect to u and ρ:

$$f(u, \rho) = \underbrace{\frac{1}{2}(\frac{u}{u_{max}})^2}_{\text{energy}} - \underbrace{\frac{u}{u_{max}}}_{\text{efficiency}} + \underbrace{\frac{\rho}{\rho_{jam}}}_{\text{safety}}. \tag{21}$$

where ρ_{jam} is the jam density. The first term represents AVs' kinetic energy. The second term denotes the driving efficiency by speed magnitude. The third term is driving safety using a congestion penalty on density ρ, implying that AVs avoid to stay in high density areas. We denote the ST-MFG as [ST-MFG-Sep]

(3) ***Non-separable***: The cost function is

$$f(u, \rho) = \underbrace{\frac{1}{2}(\frac{u}{u_{max}})^2}_{\text{energy}} - \underbrace{\frac{u}{u_{max}}}_{\text{efficiency}} + \underbrace{\frac{u\rho}{u_{max}\rho_{jam}}}_{\text{safety}}. \tag{22}$$

The difference between non-separable and separable costs is the cross product of density and velocity. It means AVs tend to decelerate in high density areas and accelerate in low density areas. We denote the ST-MFG as [ST-MFG-Non-Sep]

Proposition 6.1. [ST-MFG-Sep] is a Monotone MFG.

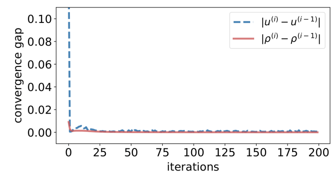
Proof. The cost function in [ST-MFG-Sep] has a separable structure (Definition 2.3):

$$\tilde{f}(u, \rho) = \frac{1}{2}(\frac{u}{u_{max}})^2 - \frac{u}{u_{max}},\ \bar{f}(u, \rho) = \frac{\rho}{\rho_{jam}}$$

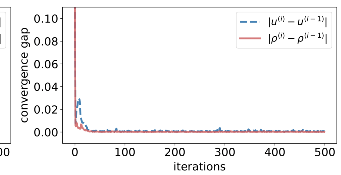
We have $\forall \rho, \rho', (\rho - \rho')(\bar{f}(u, \rho) - \bar{f}(u, \rho')) = \frac{1}{\rho_{jam}}(\rho - \rho')^2 \geqslant 0$. Therefore, Proposition 6.1 holds. Note that both [ST-MFG-LWR] and [ST-MFG-Non-Sep] contain the cross product of u and ρ, which do not satisfy the separable structure defined in Monotone MFG. □

6.2 Numerical Results

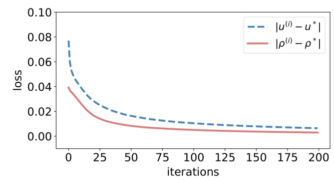
Figure 4 demonstrates the algorithm performance to solve [ST-MFG-LWR]. We assume $U(\rho) = 1 - \rho$. Solution granularity is $\Delta x = \Delta t = 10^{-3}$. The x-axis represents the iteration index during the training. Figure 4a and 4b plot the convergence gap (i.e., $|\rho^{(i)} - \rho^{(i-1)}|$, $|u^{(i)} - u^{(i-1)}|$). MFE solved by the numerical method (Section 2.4) is used as our benchmark.



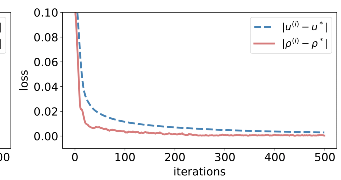
(a) Convergence gap (Pure-PIDL)



(b) Convergence gap (RL-PIDL)

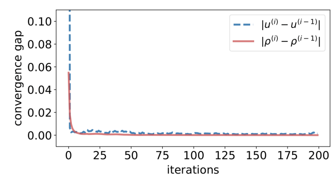


(c) Loss (Pure-PIDL)

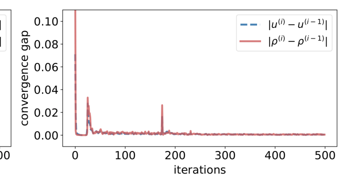


(d) Loss (RL-PIDL)

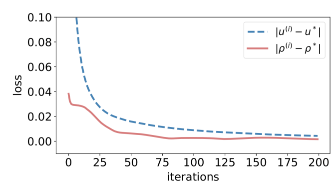
Figure 4: Algorithm performance on [ST-MFG-LWR]



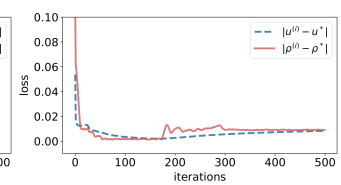
(a) Convergence gap (Pure-PIDL)



(b) Convergence gap (RL-PIDL)

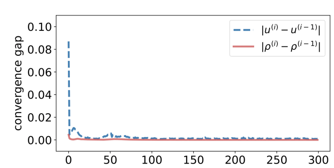


(c) Loss (Pure-PIDL)

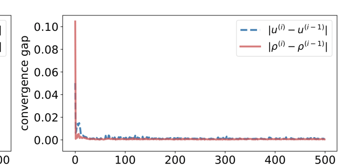


(d) Loss (RL-PIDL)

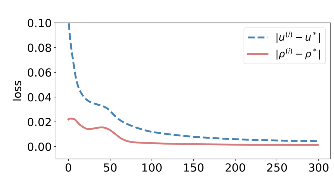
Figure 5: Algorithm performance on [ST-MFG-Sep]



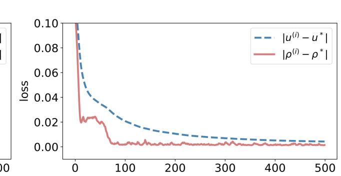
(a) Convergence gap (Pure-PIDL)



(b) Convergence gap (RL-PIDL)



(c) Loss (Pure-PIDL)

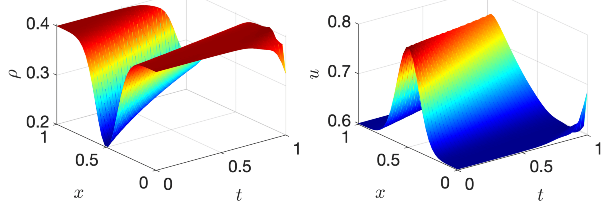


(d) Loss (RL-PIDL)

Figure 6: Algorithm performance on [ST-MFG-Non-Sep]

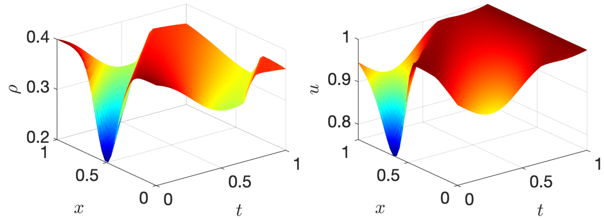
Figure 4c and 4d visualize the closeness between the benchmark and results at each iteration (i.e., $|\rho^{(i)} - \rho^*|$, $|u^{(i)} - u^*|$). Figure 5 demonstrates the algorithm performance to solve [ST-MFG-Sep]. Parameters in the cost function are: $u_{max} = 1, \rho_{jam} = 1$. Solution granularity is $\Delta x = \Delta t = 10^{-3}$. Figure 6 demonstrates the algorithm performance on [ST-MFG-Non-Sep]. Parameters remain the same as [ST-MFG-Sep]. It is shown that the pure PIDL method is faster to train than the RL-PIDL method.

Figures 7, 8, 9 demonstrate MFE (ρ^*, u^*) of [ST-MFG-LWR], [ST-MFG-Sep] and [ST-MFG-Non-Sep], respectively. The x-axis represents position x and the y-axis represents t. Compared to [ST-MFG-LWR], the initial density in [ST-MFG-Sep] and [ST-MFG-Non-Sep] quickly dissipate. The density of [ST-MFG-Non-Sep] keeps smooth and no wave forms.



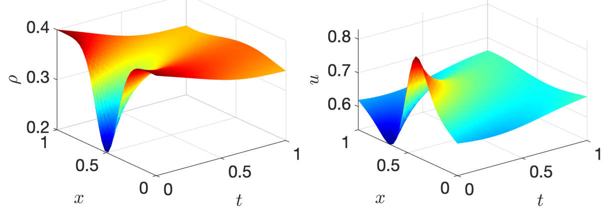
(a) Density-ρ (b) Speed-u

Figure 7: MFE of [ST-MFG-LWR]



(a) Density-ρ (b) Speed-u

Figure 8: MFE of [ST-MFG-Sep]



(a) Density-ρ (b) Speed-u

Figure 9: MFE of [ST-MFG-Non-Sep]

In Table 1, we make a comparison of different methods with two solution granularities: $\Delta t = \Delta x = 10^{-3}$ and $\Delta t = \Delta x = 10^{-6}$. The computational time of our learning methods is the training time. The numerical method does not work on ST-MFGs with solution granularity $\Delta x = \Delta t = 10^{-6}$ because the size of state space $\mathcal{X} \times \mathcal{T}$ becomes $10^6 \cdot 10^6$. Our methods provide MFEs with better resolution in time and space.

7 CONCLUSION

In this study, we establish a hybrid framework of RL and PIDL to learn MFGs, which has a generalization capability to handle large multi-agent systems in engineering and robotics application. We propose two methods: RL-PIDL and pure PIDL, and develop algorithms to solve ST-MFGs. Our methods are applied to Monotone and Non-monotone MFGs in autonomous driving systems. The overall findings include: (1) The joint framework of RL and PIDL can be replaced by the pure PIDL framework when the dynamics in the environment are known. The pure PIDL method is faster to train than the RL-PIDL method. (2) Both learning frameworks can handle ST-MFGs with finer solution granularity while numerical methods cannot. Our methods provide MFE with a satisfactory resolution in time and space.

$\Delta t, \Delta x$	Cost	Metric		**Numerical**	**PIDL**	**RL + PIDL**
$\frac{1}{10^3}$	LWR	Time		10.23	39.17	173.06
		Iterations		80	200	500
		Gap	u	$4.37 \cdot 10^{-6}$	$4.98 \cdot 10^{-4}$	$6.16 \cdot 10^{-4}$
			ρ	$2.59 \cdot 10^{-16}$	$1.35 \cdot 10^{-4}$	$2.04 \cdot 10^{-4}$
		Loss	u	-	$6.44 \cdot 10^{-3}$	$2.94 \cdot 10^{-3}$
			ρ		$2.97 \cdot 10^{-3}$	$4.01 \cdot 10^{-4}$
	Non-sep	Time		12.64	85.93	193.12
		Iterations		80	300	500
		Gap	u	$1.26 \cdot 10^{-6}$	$1.02 \cdot 10^{-3}$	$7.85 \cdot 10^{-4}$
			ρ	$1.76 \cdot 10^{-16}$	$1.25 \cdot 10^{-4}$	$4.57 \cdot 10^{-4}$
		Loss	u	-	$4.36 \cdot 10^{-4}$	$4.29 \cdot 10^{-3}$
			ρ		$1.37 \cdot 10^{-3}$	$1.46 \cdot 10^{-3}$
	Sep	Time		11.57	57.66	171.42
		Iterations		80	200	500
		Gap	u	$3.98 \cdot 10^{-6}$	$1.46 \cdot 10^{-3}$	$4.11 \cdot 10^{-4}$
			ρ	$2.37 \cdot 10^{-16}$	$2.39 \cdot 10^{-4}$	$4.75 \cdot 10^{-4}$
		Loss	u	-	$4.28 \cdot 10^{-3}$	$8.30 \cdot 10^{-3}$
			ρ		$1.59 \cdot 10^{-3}$	$9.06 \cdot 10^{-3}$
$\frac{1}{10^6}$	LWR	Time		NA	59.46	181.21
		Iterations			200	500
		Gap	u		$1.19 \cdot 10^{-3}$	$1.73 \cdot 10^{-3}$
			ρ		$7.84 \cdot 10^{-5}$	$5.83 \cdot 10^{-4}$
		Loss			-	
	Non-sep	Time			91.15	197.89
		Iterations			300	500
		Gap	u		$3.08 \cdot 10^{-3}$	$1.31 \cdot 10^{-3}$
			ρ		$6.72 \cdot 10^{-4}$	$4.89 \cdot 10^{-4}$
		Loss			-	
	Sep	Time			70.08	186.06
		Iterations			200	500
		Gap	u		$2.19 \cdot 10^{-3}$	$6.10 \cdot 10^{-3}$
			ρ		$3.52 \cdot 10^{-4}$	$1.28 \cdot 10^{-3}$
		Loss			-	

Table 1: Comparison of different methods

Our methods can be extended in many ways for future work: First, the flexibility of our PIDL module provides a more efficient learning scheme in optimal control problems where agent or system dynamics can be captured by physical rules. Second, we will study how to apply proposed learning frameworks to non-stationary MFGs with infinite time horizon.

ACKNOWLEDGMENTS

This work is partially supported by the NSF CMMI-1943998.

REFERENCES

[1] Yves Achdou, Fabio Camilli, and Italo Capuzzo-Dolcetta. 2012. Mean field games: numerical methods for the planning problem. *SIAM Journal on Control and Optimization* 50, 1 (2012), 77–109.

[2] Yves Achdou and Mathieu Laurière. 2020. *Mean Field Games and Applications: Numerical Aspects*. Springer International Publishing, Cham, 249–307.

[3] Yves Achdou and Victor Perez. 2012. Iterative strategies for solving linearized discrete mean field games systems. *Networks Heterog. Media* 7 (2012), 197–217.

[4] Berkay Anahtarc, Can Kariksiz, and Naci Saldi. 2023. Q-Learning in Regularized Mean-field Games. *Dynamic Games and Applications* 13, 1 (2023), 89–117.

[5] Dario Bauso, Xuan Zhang, and Antonis Papachristodoulou. 2017. Density Flow in Dynamical Networks via Mean-Field Games. *IEEE Trans. Automat. Control* 62, 3 (2017), 1342–1355.

[6] Jean-David Benamou and Guillaume Carlier. 2015. Augmented Lagrangian methods for transport optimization, mean field games and degenerate elliptic equations. *Journal of Optimization Theory and Applications* 167, 1 (2015), 1–26.

[7] Theophile Cabannes, Mathieu Laurière, Julien Perolat, Raphael Marinier, Sertan Girgin, Sarah Perrin, Olivier Pietquin, Alexandre M. Bayen, Eric Goubault, and Romuald Elie. 2022. Solving N-Player Dynamic Routing Games with Congestion: A Mean-Field Approach. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*.

[8] Dan Calderone and S. Shankar Sastry. 2017. Markov Decision Process Routing Games. In *2017 ACM/IEEE 8th International Conference on Cyber-Physical Systems (ICCPS)*.

[9] Pierre Cardaliaguet. 2010. *Notes on mean field games.* Technical Report. Stanford University.

[10] Pierre Cardaliaguet. 2015. *Weak solutions for first order mean field games with local coupling*. Springer, Cham, 111–158.

[11] Pierre Cardaliaguet and Saeed Hadikhanloo. 2015. Learning in Mean Field Games: the Fictitious Play. *ESAIM: Control, Optimisation and Calculus of Variations* 23 (07 2015). https://doi.org/10.1051/cocv/2016004

[12] René Carmona and Mathieu Laurière. 2021. Convergence Analysis of Machine Learning Algorithms for the Numerical Solution of Mean Field Control and Games I: The Ergodic Case. *SIAM J. Numer. Anal.* 59, 3 (2021), 1455–1485.

[13] Geoffroy Chevalier, Jerome Le Ny, and Roland Malhamé. 2015. A micro-macro traffic model based on mean-field games. In *2015 American Control Conference (ACC)*. IEEE.

[14] Yat Tin Chow, Wuchen Li, Stanley Osher, and Wotao Yin. 2019. Algorithm for Hamilton–Jacobi Equations in Density Space Via a Generalized Hopf Formula. *J. Sci. Comput.* 80, 2 (Aug 2019), 1195–1239.

[15] Romain Couillet, Samir M Perlaza, Hamidou Tembine, and Mérouane Debbah. 2012. Electrical vehicles in the smart grid: A mean field game analysis. *IEEE Journal on Selected Areas in Communications* 30, 6 (2012), 1086–1096.

[16] Romuald Elie, Julien Perolat, Mathieu Laurière, Matthieu Geist, and Olivier Pietquin. 2020. On the convergence of model free learning in mean field games. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[17] Matthieu Geist, Julien Pérolat, Mathieu Laurière, Romuald Elie, Sarah Perrin, Oliver Bachem, Rémi Munos, and Olivier Pietquin. 2022. Concave Utility Reinforcement Learning: The Mean-Field Game Viewpoint. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*.

[18] Maximilien Germain, Joseph Mikael, and Xavier Warin. 2022. Numerical resolution of McKean-Vlasov FBSDEs using neural networks. *Methodology and Computing in Applied Probability* 24, 4 (2022), 1–30.

[19] Xin Guo, Anran Hu, Renyuan Xu, and Junzi Zhang. 2019. Learning Mean-Field Games. In *Advances in Neural Information Processing Systems*, Vol. 32.

[20] Saeed Hadikhanloo. 2017. *Learning in anonymous nonatomic games with applications to first-order mean field games.*

[21] Vincent Hsiao and Dana Nau. 2022. A Mean Field Game Model of Spatial Evolutionary Games. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*.

[22] Kuang Huang, Xu Chen, Xuan Di, and Qiang Du. 2021. Dynamic driving and routing games for autonomous vehicles on networks: A mean field game approach. *Transportation Research Part C: Emerging Technologies* 128 (2021), 103189.

[23] Kuang Huang, Xuan Di, Qiang Du, and Xi Chen. 2019. Stabilizing Traffic via Autonomous Vehicles: A Continuum Mean Field Game Approach. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*.

[24] Kuang Huang, Xuan Di, Qiang Du, and Xi Chen. 2020. A game-theoretic framework for autonomous vehicles velocity control: Bridging microscopic differential games and macroscopic mean field games. *Discrete and Continuous Dynamical Systems - B* 25, 12 (2020), 4869–4903.

[25] Kuang Huang, Xuan Di, Qiang Du, and Xi Chen. 2020. Scalable traffic stability analysis in mixed-autonomy using continuum models. *Transportation Research Part C: Emerging Technologies* 111 (2020), 616–630.

[26] Minyi Huang, Roland P Malhamé, and Peter E Caines. 2006. Large population stochastic dynamic games: closed-loop McKean-Vlasov systems and the Nash certainty equivalence principle. *Communications in Information & Systems* 6, 3 (2006), 221–252.

[27] Arman C. Kizilkale and Peter E. Caines. 2013. Mean Field Stochastic Adaptive Control. *IEEE Trans. Automat. Control* 58, 4 (2013), 905–920. https://doi.org/10.1109/TAC.2012.2228032

[28] Aimé Lachapelle and Marie-Therese Wolfram. 2011. On a mean field game approach modeling congestion and aversion in pedestrian crowds. *Transportation research part B: methodological* 45, 10 (2011), 1572–1589.

[29] Jean-Michel Lasry and Pierre-Louis Lions. 2007. Mean field games. *Japanese journal of mathematics* 2, 1 (2007), 229–260.

[30] Mathieu Laurière, Sarah Perrin, Matthieu Geist, and Olivier Pietquin. 2022. *Learning Mean Field Games: A Survey.*

[31] Mathieu Lauriere, Sarah Perrin, Sertan Girgin, Paul Muller, Ayush Jain, Theophile Cabannes, Georgios Piliouras, Julien Perolat, Romuald Elie, Olivier Pietquin, and Matthieu Geist. 2022. Scalable Deep Reinforcement Learning Algorithms for Mean Field Games. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*.

[32] David Mguni, Joel Jennings, and Enrique Munoz de Cote. 2018. Decentralised learning in systems with many, many strategic agents. In *Thirty-second AAAI conference on artificial intelligence.*

[33] Paul Muller, Mark Rowland, Romuald Elie, Georgios Piliouras, Julien Perolat, Mathieu Lauriere, Raphael Marinier, Olivier Pietquin, and Karl Tuyls. 2022. Learning Equilibria in Mean-Field Games: Introducing Mean-Field PSRO. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems.*

[34] Julien Pérolat, Sarah Perrin, Romuald Elie, Mathieu Laurière, Georgios Piliouras, Matthieu Geist, Karl Tuyls, and Olivier Pietquin. 2022. Scaling Mean Field Games by Online Mirror Descent. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS '22).*

[35] Sarah Perrin, Mathieu Laurière, Julien P'erolat, Romuald Elie, Matthieu Geist, and Olivier Pietquin. 2022. Generalization in Mean Field Games by Learning Master Policies. In *AAAI Conference on Artificial Intelligence.*

[36] Sarah Perrin, Mathieu Laurière, Julien Pérolat, Matthieu Geist, Romuald Élie, and Olivier Pietquin. 2021. Mean Field Games Flock! The Reinforcement Learning Way. In *International Joint Conference on Artificial Intelligence (IJCAI-21).*

[37] Sarah Perrin, Julien Perolat, Mathieu Laurière, Matthieu Geist, Romuald Elie, and Olivier Pietquin. 2020. Fictitious Play for Mean Field Games: Continuous Time Analysis and Applications. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20).*

[38] Lars Ruthotto, Stanley J. Osher, Wuchen Li, Levon Nurbekyan, and Samy Wu Fung. 2020. A machine learning framework for solving high-dimensional mean field game and mean field control problems. *Proceedings of the National Academy of Sciences* 117, 17 (2020), 9183–9193.

[39] Rongye Shi, Zhaobin Mo, and Xuan Di. 2021. Physics-Informed Deep Learning for Traffic State Estimation: A Hybrid Paradigm Informed By Second-Order Traffic Models. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 1 (May 2021), 540–547.

[40] Justin Sirignano and Konstantinos Spiliopoulos. 2018. DGM: A deep learning algorithm for solving partial differential equations. *J. Comput. Phys.* 375 (2018), 1339–1364.

[41] Jayakumar Subramanian and Aditya Mahajan. 2019. Reinforcement Learning in Stationary Mean-Field Games. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '19).*

[42] Sriram Subramanian, Matthew Taylor, Mark Crowley, and Pascal Poupart. 2022. Decentralized Mean Field Games. In *Proceedings of the AAAI Conference on Artificial Intelligence.*

[43] E Weinan, Jiequn Han, and Arnulf Jentzen. 2017. Deep Learning-Based Numerical Methods for High-Dimensional Parabolic Partial Differential Equations and Backward Stochastic Differential Equations. *Communications in Mathematics and Statistics* 5 (2017), 349–380.

[44] Qiaomin Xie, Zhuoran Yang, Zhaoran Wang, and Andreea Minca. 2021. Learning While Playing in Mean-Field Games: Convergence and Optimality. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139).*

[45] Thaleia Zariphopoulou Xin Guo, Renyuan Xu. 2022. Entropy Regularization for Mean Field Games with Learning. In *Mathematics of Operations Research.*

[46] Jiachen Yang, Xiaojing Ye, Rakshit Trivedi, Huan Xu, and Hongyuan Zha. 2018. Deep Mean Field Games for Learning Optimal Behavior Policy of Large Populations. In *International Conference on Learning Representations.*

[47] Huibing Yin, Prashant G. Mehta, Sean P. Meyn, and Uday V. Shanbhag. 2014. Learning in Mean-Field Games. *IEEE Trans. Automat. Control* 59, 3 (2014), 629–644. https://doi.org/10.1109/TAC.2013.2287733

[48] Zihe Zhang, Lixin Li, Wei Liang, Hao Li, Ang Gao, Wei Chen, and Zhu Han. 2018. Downlink Interference Management in Dense Drone Small Cells Networks Using Mean-Field Game Theory. In *International Conference on Wireless Communications and Signal Processing.*