Decentralized Stochastic Control with Finite-Dimensional Memories: A Memory Limitation Approach

Takehiro Tottori 1, * and Tetsuya J. Kobayashi 1, 2, 3, 4

1 Department of Mathematical Informatics, Graduate School of Information Science and Technology, The University of Tokyo, Tokyo 113-8654, Japan
2 Institute of Industrial Science, The University of Tokyo, Tokyo 153-8505, Japan
3 Department of Electrical Engineering and Information Systems, Graduate School of Engineering, The University of Tokyo, Tokyo 113-8654, Japan
4 Universal Biology Institute, The University of Tokyo, Tokyo 113-8654, Japan

* Correspondence: takehiro_tottori@sat.t.u-tokyo.ac.jp

Abstract: Decentralized stochastic control (DSC) is a stochastic optimal control problem consisting of multiple controllers. DSC assumes that each controller is unable to accurately observe the target system and the other controllers. This setup results in two difficulties in DSC; one is that each controller has to memorize the infinite-dimensional observation history, which is not practical, because the memory of the actual controllers is limited. The other is that the reduction of infinite-dimensional sequential Bayesian estimation to finite-dimensional Kalman filter is impossible in general DSC, even for linear-quadratic-Gaussian (LQG) problems. In order to address these issues, we propose an alternative theoretical framework to DSC—memory-limited DSC (ML-DSC). ML-DSC explicitly formulates the finite-dimensional memories of the controllers. Each controller is jointly optimized to compress the infinite-dimensional observation history into the prescribed finite-dimensional memory and to determine the control based on it. Therefore, ML-DSC can be a practical formulation for actual memory-limited controllers. We demonstrate how ML-DSC works in the LQG problem. The conventional DSC cannot be solved except in the special LQG problems where the information the controllers have is independent or partially nested. We show that ML-DSC can be solved in more general LQG problems where the interaction among the controllers is not restricted.

Keywords: decision-making; optimal control; stochastic control; multi-agent system; decentralized control; memory limitation; mean-field control

1. Introduction

Optimal control problems of a stochastic dynamical system by decentralized multiple controllers appear in various practical applications, including real-time communication [1, 2], decentralized hypothesis testing [3], and networked control [4]. Such problems have been extensively studied in stochastic optimal control theory as decentralized stochastic control (DSC) [5–10]. DSC consists of a target system and multiple controllers (Figure 1a) and assumes that each controller cannot completely observe the state of the system and the controls of the other controllers. The information of the target system and the other controllers was only obtained via their noisy observations. Thus, each controller should be optimized to determine its control solely from its own observation history. Even for a pair of finite-dimensional state and observation, the observation history is infinite-dimensional. As a result, ideally, the optimal controller obtained theoretically should possess infinite-dimensional memory. In practical applications, however, the available memory size of each controller is finite-dimensional and often severely limited. Thus, we have to obtain the solutions based on finite-dimensional memory by employing approximation methods heuristically, which may impair the optimality of the ideal solution, especially when the available memory size is not sufficient.
**Figure 1.** Schematic diagram of (a) decentralized stochastic control (DSC) and (b) memory-limited DSC (ML-DSC). (a) DSC consists of a system and $N$ controllers. $x_t \in \mathbb{R}^d$ is the state of the target system at time $t \in [0,T]$. $y^i_t \in \mathbb{R}^{d_i}$, $y^i_{0:t} := \{y^i_\tau | \tau \in [0,t]\}$, and $u^i_t \in \mathbb{R}^{d_i}$ are the observation, the observation history, and the control of controller $i$, respectively. The controller $i$ cannot accurately observe the state of the system $x_t$ and the controls of the other controllers $u^j_t$ ($j \neq i$). It can only obtain their noisy observation $y^i_t$. Then, the controller $i$ determines the control $u^i_t$ based on the noisy observation history $y^i_{0:t}$. (b) ML-DSC explicitly formulates the finite-dimensional memory $z^i_t \in \mathbb{R}^{d_i}$. Controller $i$ compresses the infinite-dimensional observation history $y^i_{0:t}$ into the finite-dimensional memory $z^i_t$ by optimally designing control over the memory $v^i_t \in \mathbb{R}^{d_i}$ as well as control over the state $u^i_t \in \mathbb{R}^{d_i}$.

Moreover, another difficulty arises in DSC due to the decentralized setting. If the number of controllers is one, or if all controllers share their observation histories, DSC is reduced to partially observable stochastic control (POSC), in which the observation histories of all controllers can be summarized optimally as the posterior probability of the state by sequential Bayesian estimation [11–16]. The posterior probability of the state is also infinite-dimensional, and thus the same problem as DSC still survives, even for POSC. Nevertheless, in POSC, this difficulty can be circumvented by focusing only on the linear-quadratic-Gaussian (LQG) setting under which the posterior probability of the state can be represented by the finite-dimensional mean vector and covariance matrix of Gaussian distribution and the sequential Bayesian estimation can be computed by the Kalman filter [11,12,14]. Therefore, POSC is practically solved, at least in the LQG problem. The difficulty in DCS is that this nice property of LQG is not retained.

In DCS, each controller cannot access the observation histories of the other controllers, as well as the state of the system, which causes each controller to estimate all the others from their own observation history. This hampers the Bayesian estimation of the posterior to be computed sequentially and prevents the infinite-dimensional observation history from being compressed into the finite-dimensional sufficient statistics, even for the LQG problem. Some theoretical studies have addressed this issue by restricting the interaction among the controllers. If the information the controllers have is independent [8–10] or partially nested [17–22], DSC can enjoy the nice property of the LQG problem and be solved explicitly and optimally with finite-dimensional memory. However, the LQG problem with more general interactions as well as non-LQG problems are still an open problem in DSC.

In order to address these issues, we propose an alternative theoretical framework to DSC, memory-limited DSC (ML-DSC), which is the decentralized version of memory-limited POSC (ML-POSC) [23,24]. The two major difficulties in DSC originate from the ignorance of constraints over controllers when we derive the optimal estimation and control solution. Unlike conventional DSC, ML-DSC explicitly formulates finite-dimensional memories of the controllers and their capacities (Figure 1b). In ML-DSC, each controller is optimized to compress the infinite-dimensional observation history into the prescribed
finite-dimensional memory and to determine the control based on it. In other words, each controller controls not only the dynamics of the target system but also the dynamics of its own memory. The formulation of ML-DSC enables us to evade the difficulties mentioned above.

Furthermore, we provide a way to solve the optimization problem associated with the ML-DSC formulation. Specifically, we address the optimization problem by converting ML-DSC in the state space into the deterministic optimal control problem in the probability density function space. This technique has recently been used in mean-field stochastic control [25,26] and ML-POSC [23,24], and is also effective for ML-DSC. Following that, we can solve ML-DSC in a similar way to the deterministic optimal control problem on the probability density function space; the optimal control function of ML-DSC was obtained by jointly solving the Hamilton–Jacobi–Bellman (HJB) equation and the Fokker–Planck (FP) equation. HJB–FP equations also appear in mean-field stochastic game and control [25–29] and ML-POSC [23,24], and numerous numerical algorithms have been proposed [24,30–32].

Using these numerical algorithms, ML-DSC may be solved effectively, even in general problems. It should be noted that a similar idea to ML-DSC was also employed in a decentralized partially observable Markov decision process (DEC-POMDP) for more than a decade [33–39]. However, the algorithms of the finite-state controller of DEC-POMDP strongly depend on the discreteness, and thus they are not applicable to ML-DSC where the continuous time and state are considered.

We applied ML-DSC and our algorithm to the LQG problem. The conventional DSC can only be solved for special LQG problems where the information of the controllers is independent [8–10] or partially nested [17–22]. In contrast, ML-DSC can be solved even in LQG problems with more general interactions among the controllers. In LQG problems of POSC, estimation and control are clearly separated, and are optimized by the Kalman filter and the Riccati equation, respectively [11,14]. In the LQG problems of ML-DSC, estimation and control are not clearly separated and are jointly optimized by the modified Riccati equation, which is called the decentralized Riccati equation in this paper. We noted that this coupling of estimation and control also appears in the conventional DSC [17–22] and ML-POSC [23,24]. Therefore, it may be induced by decentralized structure and memory limitation. Finally, we conducted two numerical experiments for the LQG problems of ML-DSC. One controls one-dimensional divergent state dynamics, and the other controls two-dimensional oscillatory state dynamics. These numerical experiments demonstrate that the decentralized Riccati equation is superior to the Riccati equation in the LQG problems of ML-DSC.

The rest of this paper is organized as follows. In Section 2, we briefly review the conventional DSC. In Section 3, we formulate ML-DSC. In Section 4, we solve ML-DSC. In Section 5, we apply ML-DSC to the LQG problem. In Section 6, we conduct the numerical experiments of two LQG problems in ML-DSC. In Section 7, we discuss this paper.

2. Review of Decentralized Stochastic Control

In this section, we briefly review the conventional DSC (Figure 1a) [8–10]. DSC consists of a target system and N controllers. Vector \( x_t \in \mathbb{R}^{d_x} \) is the state of the system at time \( t \in [0, T] \), which evolves by the following stochastic differential equation (SDE):

\[
 dx_t = b(t, x_t, u_t)dt + \sigma(t, x_t, u_t)d\omega_t,
\]

where \( x_0 \) obeys \( p_0(x_0) \), \( \omega_t \in \mathbb{R}^{d_\omega} \) is the standard Wiener process, \( u^i_t \in \mathbb{R}^{d_u} \) is the control of controller \( i \), and \( u_t := (u^1_t, u^2_t, \ldots, u^N_t) \in \mathbb{R}^{d_u} \) is the joint control of all controllers. We noted that \( d_u := \sum_{i=1}^{N} d^i_u \). DSC often assumes that the system is composed of \( N \) agents and that the state of the system is decomposed into \( x_t := (x^1_t, x^2_t, \ldots, x^N_t) \in \mathbb{R}^{d_x} \) where \( x^i_t \in \mathbb{R}^{d^{i}_x} \) is the state of agent \( i \). In this paper, we do not assume such a situation, because our formulation of the state of the system includes it as a special case.
In DSC, controller $i$ cannot completely observe the state of the system $x_t$ and the joint control of all controllers $u_t$. It can only obtain the noisy observation $y^i_t \in \mathbb{R}^{d_y}$, which evolves by the following SDE:

$$
dy^i_t = h^i(t, x_t, u_t)dt + \gamma^i(t, x_t, u_t)d\nu^i_t, $$

(2)

where $y^i_0$ obeys $p^i_0(y^i_0)$, and $\nu^i_t \in \mathbb{R}^{d_\nu}$ is the standard Wiener process. Controller $i$’s observation $y^i_t$ is controlled by the other controllers through the joint control $u_t$, which expresses the communication among the controllers. Controller $i$ determines its control $u^i_t$ based on the observation history $y^i_{0:t} := \{y^i_t | t \in [0, T]\}$ as follows:

$$
u^i_t = u^i(t, y^i_{0:t}). $$

(3)

The objective function of DSC is given by the following expected cumulative cost function:

$$
J[u] := \mathbb{E}_{p(x_t; y^i_{0:T}; u)} \left[ \int_0^T f(t, x_t, u_t)dt + g(x_T) \right],
$$

(4)

where $f$ is the running cost function, $g$ is the terminal cost function, $p(x_t; y^i_{0:T}; u)$ is the joint probability of $x_{0:T}$ and $y^i_{0:T}$ given that $u$ is a parameter, and $\mathbb{E}_{p[\cdot]}$ is the expectation with respect to $p$. DSC is the problem to find the optimal joint control function $u^*$ that minimizes the objective function $J[u]$:

$$
u^* := \arg \min_u J[u]. $$

(5)

In DSC, controller $i$ needs to memorize the infinite-dimensional observation history $y^i_{0:t}$ to determine the optimal control $u^i = u^i(t, y^i_{0:t})$. This is one of the major obstacles in DSC for implementing controllers with finite and limited memory.

3. Memory-Limited Decentralized Stochastic Control

In this section, we formulate ML-DSC, which can circumvent the difficulty in DSC by explicitly formulating finite-dimensional memory of the controllers.

3.1. Problem Formulation

In this subsection, we formulate ML-DSC (Figure 1b). ML-DSC explicitly formulates the finite-dimensional memory of controller $i$ by $z^i_t \in \mathbb{R}^{d_z}$. The memory dimension $d_z$ is prescribed by the available memory size of the controller $i$. The controller $i$ compresses the infinite-dimensional observation history $y^i_{0:t}$ into the finite-dimensional memory $z^i_t$ by the following SDE:

$$
dz^i_t = c^i(t, z^i_t, v^i_t)dt + \kappa^i(t, z^i_t, v^i_t)dy^i_t + \eta^i(t, z^i_t, v^i_t)d\xi^i_t, $$

(6)

where $z^i_0$ obeys $p^i_0(z^i_0)$, $\xi^i_t \in \mathbb{R}^{d_\xi}$ is the standard Wiener process, and $v^i_t \in \mathbb{R}^{d_v}$ is the control of the memory. Unlike the conventional DSC, ML-DSC can take into account the intrinsic stochasticity of the memory, which is modeled by the standard Wiener process $d\xi^i_t$ in the memory dynamics (6). In addition, the compression of the infinite-dimensional observation history $y^i_{0:t}$ into the finite-dimensional memory $z^i_t$ is optimized by the memory control $v^i_t$. In ML-DSC, the controller $i$ determines the state control $u^i_t$ and the memory control $v^i_t$ based on the finite-dimensional memory $z^i_t$ as follows:

$$
u^i_t = u^i(t, z^i_t), \quad v^i_t = v^i(t, z^i_t). $$

(7)
We expect that the optimal memory control function \( f \) where \( d v \) is the optimal memory control function of conventional stochastic optimal control problems. Furthermore, it clarifies the difference between ML-DSC and the state. This generalization is useful for mathematical investigations by simplifying the scope of this work.

In this subsection, we generalize the formulation of ML-DSC based on the extended state. This generalization is useful for mathematical investigations by simplifying the notation of ML-DSC. Furthermore, it clarifies the difference between ML-DSC and the conventional DSC.

In ML-DSC, controller \( i \) determines the optimal control functions \( u^*_i \) and \( v^*_i \) based only on the finite-dimensional memory \( z^*_i \). In addition, ML-DSC can take into account the intrinsic stochasticity and the control cost of the memory. Thus, ML-DSC can explicitly accommodate various realistic constrains of the controllers such as memory size, noise in the controllers, and cost for updating memory, none of which can be explicitly addressed in the conventional DSC.

It should be noted that here we consider memory size only for storing continuous time-series with finite-dimensional vectors. While memory size also matters when we consider quantization and storing of real valued observations, these topics are out of the scope of this work.

### 3.2. Extended State

In ML-DSC, controller \( i \) determines the optimal control functions \( u^*_i \) and \( v^*_i \) based only on the finite-dimensional memory \( z^*_i \). In addition, ML-DSC can take into account the intrinsic stochasticity and the control cost of the memory. Thus, ML-DSC can explicitly accommodate various realistic constrains of the controllers such as memory size, noise in the controllers, and cost for updating memory, none of which can be explicitly addressed in the conventional DSC.

It should be noted that here we consider memory size only for storing continuous time-series with finite-dimensional vectors. While memory size also matters when we consider quantization and storing of real valued observations, these topics are out of the scope of this work.

The objective function of ML-DSC is given by the following expected cumulative cost function:

\[
J[u, v] := \mathbb{E}_p(x_{0:T}, y_{0:T}, z_{0:T}; u, v) \int_0^T f(t, x_t, u_t, v_t) dt + g(x_T),
\]

where \( f \) is the running cost function, \( g \) is the terminal cost function, \( p(x_{0:T}, y_{0:T}, z_{0:T}; u, v) \) is the joint probability of \( x_{0:T}, y_{0:T} \) and \( z_{0:T} \) given \( u \) and \( v \) as parameters, and \( \mathbb{E}_p[\cdot] \) is the expectation with respect to \( p \). Unlike the cost function \( f \) of DSC in Equation (4), the cost function \( f \) of ML-DSC in Equation (8) depends on the memory control \( v_t \) as well as the state control \( u_t \). From a practical viewpoint, it should be natural to consider both costs of control and memory. ML-DSC optimizes the state control function \( u_t \) and the memory control function \( v_t \) based on the objective function \( J[u, v] \):

\[
u^*_t, v^*_t := \arg \min_{u, v} J[u, v].
\]

The optimal memory control function \( v^* := (v^*_1, v^*_2, \ldots, v^*_N) \) optimizes the memory dynamics (6), which can be interpreted as the optimization of the compression of the observation history into the finite-dimensional memory. In the LQG problem of POSC, the optimal memory control function \( v^* \) makes the memory dynamics into the Kalman filter, which is the optimal compression of the observations history in this problem [23]. We expect that the optimal memory control function \( v^* \) is also effective for more general problems of ML-DSC.

In ML-DSC, controller \( i \) determines the optimal control functions \( u^*_i \) and \( v^*_i \) based only on the finite-dimensional memory \( z^*_i \). In addition, ML-DSC can take into account the intrinsic stochasticity and the control cost of the memory. Thus, ML-DSC can explicitly accommodate various realistic constrains of the controllers such as memory size, noise in the controllers, and cost for updating memory, none of which can be explicitly addressed in the conventional DSC.

It should be noted that here we consider memory size only for storing continuous time-series with finite-dimensional vectors. While memory size also matters when we consider quantization and storing of real valued observations, these topics are out of the scope of this work.

### 3.2. Extended State

In this subsection, we generalize the formulation of ML-DSC based on the extended state. This generalization is useful for mathematical investigations by simplifying the notation of ML-DSC. Furthermore, it clarifies the difference between ML-DSC and the conventional stochastic optimal control problems.

We define the extended state \( s_t \in \mathbb{R}^{d_s} \), the extended control \( \hat{u}_t^i \in \mathbb{R}^{d_u^i} \), the extended joint control \( \hat{u}_t \in \mathbb{R}^{d_u} \), and the extended standard Wiener process \( \hat{\omega}_t \in \mathbb{R}^{d_{\omega}} \) as follows:

\[
s_t := \begin{pmatrix} x_t \\ z^1_t \\ \vdots \\ z^N_t \end{pmatrix}, \quad \hat{u}_t^i := \begin{pmatrix} u^1_t \\ u^2_t \\ \vdots \\ u^N_t \end{pmatrix}, \quad \hat{u}_t := \begin{pmatrix} \hat{u}_1^1 \\ \hat{u}_2^1 \\ \vdots \\ \hat{u}_N^1 \end{pmatrix}, \quad \hat{\omega}_t := \begin{pmatrix} \hat{\omega}_1^t \\ \hat{\omega}_2^t \\ \vdots \\ \hat{\omega}_N^t \end{pmatrix},
\]

where \( d_s := d_x + \sum_{i=1}^N d^1_i \), \( d^1_i := d^1_{u^i} + d^1_{v^i} \), \( d_a := \sum_{i=1}^N d^1_{u^i} \), and \( d_{\omega} := d_{\omega} + \sum_{i=1}^N d^1_{v^i} \).
Based on the extended state $s_t$, the extended joint control $\tilde{u}_i$, and the extended standard Wiener process $\tilde{\omega}_t$, the state, observation, and memory SDEs, i.e., Equations (1), (2), and (6), are summarized as follows:

$$
\begin{align*}
    ds_t &= \left( \begin{array}{c}
        b \\
        c^1 + \kappa^1 h^1 \\
        \vdots \\
        c^N + \kappa^N h^N
    \end{array} \right) dt + \left( \begin{array}{cccc}
        \sigma & O & \cdots & O \\
        O & \kappa^1 \gamma^1 & \cdots & O \\
        \vdots & \vdots & \ddots & \vdots \\
        O & O & \cdots & \kappa^N \gamma^N
    \end{array} \right) d\tilde{\omega}_t, \\
    =: \hat{b}(t, s_t, \tilde{u}_i) d\tilde{\omega}_t,
\end{align*}
$$

where $p_0(s_0) = p_0(x_0) \prod_{i=1}^{N} p_0(z_i^0)$. Thus, the SDE of ML-DSC can be generalized as follows:

$$
\begin{align*}
    ds_t &= \hat{b}(t, s_t, \tilde{u}_i) dt + \tilde{\sigma}(t, s_t, \tilde{u}_i) d\tilde{\omega}_t,
\end{align*}
$$

where $s_0$ obeys $p_0(s_0)$. We note that the structures of $\hat{b}(t, s_t, \tilde{u}_i)$ and $\tilde{\sigma}(t, s_t, \tilde{u}_i)$ in Equation (12) are not necessarily restricted to those in Equation (11). Importantly, in ML-DSC, controller $i$ determines the extended control $\tilde{u}_i^t$ based solely on the memory $z_i^t$ as follows:

$$
\tilde{u}_i^t = \tilde{u}^t(t, z_i^t).
$$

The objective function of ML-DSC (8) is generalized as follows:

$$
\begin{align*}
    J[\tilde{u}] := E_{p(s_{0:T}) \tilde{\omega}} \left[ \int_0^T \hat{f}(t, s_t, \tilde{u}_i) dt + \tilde{g}(s_T) \right],
\end{align*}
$$

where $\hat{f}$ is the running cost function and $\tilde{g}$ is the terminal cost function. Therefore, the generalized ML-DSC is the problem to find the optimal extended joint control function $\tilde{u}^*$ that minimizes the objective function $J[\tilde{u}]$:

$$
\tilde{u}^* := \arg \min_{\tilde{u}} J[\tilde{u}]
$$

under the constraint of Equation (13).

This generalization (12)–(15) clarifies the difference between ML-DSC and the conventional stochastic optimal control problems. If controller $i$ determines the extended control $\tilde{u}_i^t$ based on the whole extended state $s_t := (x_t, z_1^t, \ldots, z_N^t)$ as $\tilde{u}_i^t = \tilde{u}^t(t, s_t)$, this problem becomes equivalent to completely observable stochastic control (COSC), which is the most basic stochastic optimal control problem (Figure 2a) [13,14,40]. Furthermore, if controller $i$ determines the extended control $\tilde{u}_i^t$ based on the joint memory $z_t := (z_1^t, \ldots, z_N^t)$ as $\tilde{u}_i^t = \tilde{u}^t(t, z_t)$, this problem is reduced to ML-POSC in which all controllers share their information (Figure 2b) [23,24]. ML-DSC determines the extended control $\tilde{u}_i^t$ based solely on its own memory $z_i^t$ as $\tilde{u}_i^t = \tilde{u}^t(t, z_i^t)$ (13), which is different from COSC and ML-POSC (Figure 2c). While ML-DSC cannot be solved in a similar way as COSC [14,40,41], as is shown in the next section, it can be solved in a similar way as ML-POSC [23,24] because the method of ML-POSC is more general than that of COSC.

In the following section, we mainly consider the formulation of this subsection rather than that of Section 3.1 because it is simpler and more general. Moreover, we omit $\omega$ for the notational simplicity.
Figure 2. Schematic diagram of (a) completely observable stochastic control (COSC), (b) memory-limited partially observable stochastic control (ML-POSC), and (c) memory-limited decentralized stochastic control (ML-DSC). Blue and gray regions indicate the observables and the unobservables for the controller 1, respectively.

4. Derivation of Optimal Control Function

In this section, we solve ML-DSC by employing the technique in mean-field stochastic control [25,26] and ML-POSC [23,24].

4.1. Derivation of Optimal Control Function

In this subsection, we derive the optimal control function of ML-DSC. In ML-DSC, each controller cannot directly access the information about the state of the system and the memories of the other controllers. This constraint makes ML-DSC unable to be solved by the conventional methods of COSC, such as Bellman’s dynamic programming principle on the extended state space [13,14,40]. In order to address this issue, we converted ML-DSC on the extended state space into the deterministic optimal control problem on the probability density function space. The similar technique has also been used in mean-field stochastic control [25,26] and ML-POSC [23,24], and it is more effective for a broader class of stochastic optimal control problems than the conventional methods of COSC.

The extended state SDE (12) can be converted into the following Fokker–Planck (FP) equation:

$$\frac{\partial p_t(s)}{\partial t} = \mathcal{L}_u^+ p_t(s),$$  

(16)

where the initial condition is given by $p_0(s)$, and $\mathcal{L}_u^+$ is the forward diffusion operator, which is defined by

$$\mathcal{L}_u^+ p_t(s) := -\sum_{i=1}^{d_x} \frac{\partial (b_i(t,s,u)p_t(s))}{\partial s_i} + \frac{1}{2} \sum_{i,j=1}^{d_x} \frac{\partial^2 (D_{ij}(t,s,u)p_t(s))}{\partial s_i \partial s_j},$$  

(17)

where $D(t,s,u) := \sigma(t,s,u)\sigma^T(t,s,u)$. The objective function (14) can be calculated as follows:

$$J[u] = \int_0^T \bar{f}(t,p_t,u_t)dt + \bar{g}(p_T),$$  

(18)

where $\bar{f}(t,p,u) := E_p[f(t,s,u)]$ and $\bar{g}(p) := E_p[g(s)]$. We note that $\tilde{\cdot}$ is omitted for the notational simplicity. From Equations (16) and (18), ML-DSC on the extended state space is converted into the deterministic optimal control problem on the probability density function space.

If being represented by the extended state, each controller cannot completely access the information of the extended state in ML-DSC, which hampers the conventional methods of COSC. By lifting the state variable from the extended state to its probability density function, such a difficulty can be avoided, because any controllers can completely access the probability density function from its deterministic nature. As a result, the optimal
The optimal control function of ML-DSC satisfies the following equation:

\[
    u^{ii}(t, z^i) = \arg \min_{u^i} \mathbb{E}_{p_t(s^{-i}|z^i)} \left[ \mathcal{H} \left( t, s, (u^{-is}, u') \right) \right], \quad \forall i \in \{1, 2, \ldots, N \},
\]

where \( \mathcal{H} \) is the Hamiltonian, which is defined as follows:

\[
    \mathcal{H}(t, s, u, w) := f(t, s, u) + L_u w(t, s),
\]

where \( L_u \) is the backward diffusion operator, which is defined as follows:

\[
    L_u w(t, s) := \sum_{i=1}^{d_i} b_i(t, s, u) \frac{\partial w(t, s)}{\partial s_i} + \frac{1}{2} \sum_{i,j=1}^{d_i} D_{ij}(t, s, u) \frac{\partial^2 w(t, s)}{\partial s_i \partial s_j},
\]

which is the conjugate of \( L_u^\dagger \) as follows:

\[
    \oint w(t, s) L_u^\dagger p(t, s) ds = \oint p(t, s) L_u w(t, s) ds.
\]

Variables \( s^{-i} \in \mathbb{R}^{d_x}, u^{-is} \in \mathbb{R}^{d_u}, \) and \( (u^{-is}, u') \in \mathbb{R}^{d_u} \) are defined as follows:

\[
    s^{-i} := \begin{pmatrix} x \\ z^1 \\ \vdots \\ z^{i-1} \\ z^{i+1} \\ \vdots \\ z^N \end{pmatrix}, \quad u^{-is} := \begin{pmatrix} u^{1s} \\ \vdots \\ u^{is-1} \\ u^{is} \\ \vdots \\ u^{N^s} \end{pmatrix}, \quad (u^{-is}, u') := \begin{pmatrix} u^{1s} \\ \vdots \\ u^{is-1} \\ u^i \\ \vdots \\ u^{N^s} \end{pmatrix},
\]

where \( d_x^{-i} := d_x - d_x^i \) and \( d_u^{-i} := d_u - d_u^i \). Function \( w(t, s) \) is the solution of the following Hamilton–Jacobi–Bellman (HJB) equation:

\[
    -\frac{\partial w(t, s)}{\partial t} = \mathcal{H}(t, s, u^s, w),
\]

where \( w(T, s) = g(s) \). Function \( p_t(s^{-i}|z^i) := p_t(s)/\oint p_t(s) ds^{-i} \) is the conditional probability density function of \( s^{-i} \) given \( z^i \), and \( p_t(s) \) is the solution of FP Equation (16) driven by \( u^s \). We note that \( \cdot \) is omitted for the notational simplicity.

**Proof.** The proof is shown in Appendix A. \( \Box \)

We note that the optimality condition (19) is a necessary condition of the optimal control function of ML-DSC, not a sufficient condition. The optimality condition (19) becomes a necessary and sufficient condition when the expected Hamiltonian \( \mathcal{H}(t, p, u, w) := \mathbb{E}_{p_t(s)}[\mathcal{H}(t, s, u, w)] \) is convex with respect to \( p \) and \( u \). This proof is almost the same with Reference [24]. In the following, the control function of ML-DSC that satisfies the optimality condition (19) is called the optimal control function of ML-DSC.

### 4.2. Numerical Algorithm

The optimal control function of ML-DSC (19) is obtained by jointly solving FP Equation (16) and HJB Equation (24). HJB-FP equations also appear in mean-field stochas-
tic game and control [25–29] and ML-POSC [23, 24], and numerous numerical algorithms have been developed [24, 32]. As a result, ML-DSC may be solved practically by employing these numerical algorithms.

One of the most basic numerical algorithms to solve HJB-FP equations is the forward-backward sweep method (the fixed-point iteration method) [24, 32, 42–44], which computes the forward FP Equation (16) and the backward HJB Equation (24) alternately. While the convergence of the forward-backward sweep method is not guaranteed in mean-field stochastic game and control [32, 42–44], it is guaranteed in ML-POSC because the coupling of HJB-FP equations is limited to the optimal control function in ML-POSC [24]. The convergence of the forward-backward sweep method is also guaranteed in ML-DSC for the same reason as ML-POSC. This proof is almost the same as Reference [24].

**4.3. Comparison with Completely Observable Stochastic Control or Memory-Limited Partially Observable Stochastic Control**

COSC and ML-POSC can be solved in a similar way to ML-DSC [23, 24]. In COSC, controller i can completely observe the state of the target system $x_t$ and the memories of the other controllers $z^j_t$ ($j \neq i$), as well as its own memory $z^i_t$ (Figure 2a) [13, 14, 40]. As a result, the control $u^i_t$ is determined based on the whole extended state $s_t := (x_t, z^1_t, \ldots, z^N_t)$ as $u^i_t = u^i(t, s_t)$. From Pontryagin’s minimum principle on the probability density function space, the optimal control function of COSC is given by the following equation:

$$u^i(t, s) = \arg\min_{u^i} \mathcal{H}\left(t, s, (u^{-i}t, u^i), w\right), \quad (25)$$

where $w(t, z)$ is the solution of HJB Equation (24). This result is the same with Bellman’s dynamic programming principle on the state space. Thus, Pontryagin’s minimum principle on the probability density function space can be interpreted as the generalization of Bellman’s dynamic programming principle on the state space.

In ML-POSC, controller i can observe the memories of the other controllers $z^j_t$ ($j \neq i$) as well as its own memory $z^i_t$ (Figure 2b) [23, 24]. As a result, the control $u^i_t$ is determined based on the joint memory $s_t := (z^1_t, \ldots, z^N_t)$ as $u^i_t = u^i(t, s_t)$. From Pontryagin’s minimum principle on the probability density function space, the optimal control function of ML-POSC is given by the following equation:

$$u^i(t, s) = \arg\min_{u^i} \mathbb{E}_{p_t(x|z)}\left[\mathcal{H}\left(t, s, (u^{-i}t, u^i), w\right)\right], \quad (26)$$

where $w(t, s)$ is the solution of HJB Equation (24). $p_t(x|z) := p_t(s) / \int p_t(s)\,dx$ is the conditional probability density function of $x$ given $z$, and $p_t(s)$ is the solution of FP Equation (16) driven by $u^*$. Although HJB Equation (24) is the same for COSC, ML-POSC, and ML-DSC, the optimal control function is different. Notably, the optimal control functions of ML-POSC and ML-DSC depend on FP Equation (16) because they need to estimate unobservables from observables. ML-POSC needs to estimate the state of the system $x_t$ from the joint memory of all controllers $z_t$. ML-DSC needs to estimate the memories of the other controllers $z^j_t$ ($j \neq i$) as well as the state of the system $x_t$ from its own memory $z^i_t$.

In COSC, the optimal control function depends only on HJB Equation (24). As a result, it can be obtained by solving the HJB Equation (24) backward in time from the terminal condition, which is called the value iteration method [45–47]. By contrast, in ML-POSC and ML-DSC, the optimal control function cannot be obtained by the value iteration method because it depends on FP Equation (16) as well as HJB Equation (24). Instead, it can be obtained by the forward-backward sweep method, which computes the forward FP Equation (16) and the backward HJB Equation (24) alternately [24].
5. Linear-Quadratic-Gaussian Problem

In this section, we demonstrate how ML-DSC works by applying it to the LQG problem. In the conventional DSC, the LQG problem can be solved only when the information of the controllers is independent [8–10] or partially nested [17–22]. By contrast, in ML-DSC, the LQG problem can be solved without restricting the interaction among the controllers.

5.1. Problem Formulation

In this subsection, we formulate the LQG problem of ML-DSC. In this problem, the extended state SDE (12) is given as follows:

\[ ds_t = (A(t)s_t + B(t)u_t)dt + \sigma(t)d\omega_t \]

\[ = \left( A(t)s_t + \sum_{i=1}^{N} B_i(t)u_i^t \right)dt + \sigma(t)d\omega_t, \]

(27)

where the initial condition is given by the Gaussian distribution \( p_0(s) := \mathcal{N}(s|\mu_0, \Sigma_0) \). Furthermore, we note that \( \omega \) is omitted for the notational simplicity. In ML-DSC, controller \( i \) determines the control \( u_i^t \) based on the memory \( z_i^t \) as follows:

\[ u_i^t = u^t(t, z_i^t). \]

(28)

In the extended state SDE (27), the interaction among the controllers is not restricted. For example, each controller is allowed to control the memories of the other controllers from its own memory through the state or the observations. In this case, the memories of the controllers do not become independent or partially nested. It becomes obvious in the numerical experiments in Section 6.

The objective function (14) is given as follows:

\[ J[u] := \mathbb{E}_{p(s_0,T,u)} \left[ \int_{0}^{T} \left( s_i^T Q(t)s_i + u_i^T R(t)u_i + s_i^T Ps_T \right) dt \right], \]

\[ = \mathbb{E}_{p(s_0,T,u)} \left[ \int_{0}^{T} \left( s_i^T Q(t)s_i + \sum_{i=1}^{N} \sum_{j=1}^{N} (u_i^t)^T R_{ij}(t)u_j^t \right) dt \right], \]

(29)

where \( Q(t) \succ 0, R(t) \succ 0, \) and \( P \succ 0 \). The objective of this problem is to find the optimal control function \( u^* \) that minimizes the objective function \( J[u] \):

\[ u^* := \arg\min_{u} J[u]. \]

(30)

In this paper, we assume that \( R(t) \) is the block diagonal matrix, i.e., the costs of different controllers are independent as follows:

\[ R(t) = \begin{pmatrix} R_{11}(t) & O & \cdots & O \\ O & R_{22}(t) & \cdots & O \\ \vdots & \vdots & \ddots & \vdots \\ O & O & \cdots & R_{NN}(t) \end{pmatrix}, \]

(31)

where \( R_{ii}(t) \succ 0 \). In this case, the objective function (29) can be calculated as follows:

\[ J[u] = \mathbb{E}_{p(s_0,T,u)} \left[ \int_{0}^{T} \left( s_i^T Q(t)s_i + \sum_{i=1}^{N} (u_i^t)^T R_{ii}(t)u_i^t \right) dt + s_i^T Ps_T \right]. \]

(32)

If this assumption does not hold, the optimal control function cannot be derived explicitly. This problem is similar with the Witsenhausen’s counterexample, which demonstrates the difficulty of DSC and is historically recognized as an important problem [5,48,49].
However, this assumption is not critical in many applications as the control cost matrix is usually diagonal.

### 5.2. Derivation of Optimal Control Function

In this subsection, we derive the optimal control function of the LQG problem of ML-DSC by applying Theorem 1. In this problem, the probability density function of the extended state $\hat{s}$ at time $t$ is given by the Gaussian distribution $p_t(s) := \mathcal{N}(s|\mu(t), \Sigma(t))$, in which $\mu(t)$ is the mean vector and $\Sigma(t)$ is the covariance matrix. Defining the stochastic extended state $\hat{s} := s - \mu$, the conditional expectation $\mathbb{E}_{p_t(s)}[s]$ can be calculated as follows:

$$
\mathbb{E}_{p_t(s)}[s] = \mu(t) + K_i(t)\hat{s},
$$

where $K_i(t) \in \mathbb{R}^{d_x \times d_z}$ is defined as follows:

$$
K_i(t) := \begin{pmatrix}
\begin{array}{cccc}
O & \cdots & O & \Sigma_{xz_i}(t)\Sigma_{z_i^{-1}}(t) & O & \cdots & O \\
O & \cdots & O & \Sigma_{z_i^{-1}}(t) & O & \cdots & O \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
O & \cdots & O & \Sigma_{z_i^{-1}}(t) & O & \cdots & O \\
O & \cdots & O & I & O & \cdots & O \\
O & \cdots & O & \Sigma_{z_i^{-1}}(t) & O & \cdots & O \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
O & \cdots & O & \Sigma_{z_i^{-1}}(t) & O & \cdots & O
\end{array}
\end{pmatrix}.
$$

Function $K_i(t)$ is the zero matrix except for the columns corresponding to $z^i$. By applying Theorem 1 to the LQG problem of ML-DSC, we obtained the following theorem:

**Theorem 2.** In the LQG problem, the optimal control function of ML-DSC satisfies the following equation:

$$
u^{k^*}(t, z^i) = -R^{-1}_{ii}B_i^T(\Psi(t) + \Phi K_i \hat{s}),$$

where $K_i(t)$ is defined by Equation (34), which depends on $\Sigma(t)$. Functions $\mu(t)$ and $\Sigma(t)$ are the solutions of the following ordinary differential equations (ODEs):

$$
\frac{d\mu}{dt} = \left(A - B R^{-1} B^T \Psi \right) \mu,
$$

$$
\frac{d\Sigma}{dt} = \sigma\sigma^T + \left(A - \sum_{i=1}^N B_i R^{-1}_{ii} B_i^T \Phi \Phi \right) \Sigma + \Sigma \left(A - \sum_{i=1}^N B_i R^{-1}_{ii} B_i^T \Phi \Phi \right)^T,
$$

where $\mu(0) = \mu_0$ and $\Sigma(0) = \Sigma_0$. Functions $\Psi(t)$ and $\Phi(t)$ are the solutions of the following ODEs:

$$
-\frac{d\Psi}{dt} = Q + A^T \Psi + \Psi A - \Psi B R^{-1} B^T \Psi,
$$

$$
-\frac{d\Phi}{dt} = Q + A^T \Phi + \Phi A - \Phi B R^{-1} B^T \Phi + \sum_{i=1}^N (I - K_i)^T \Phi B_i R^{-1}_{ii} B_i^T \Phi (I - K_i),
$$

where $\Psi(T) = \Phi(T) = P$.

**Proof.** The proof is shown in Appendix B. \square

In the LQG problem of ML-DSC, FP Equation (16) is reduced to Equations (36) and (37), and HJB Equation (24) is reduced to Equations (38) and (39). The optimal control function (35)
is decomposed into the deterministic part and the stochastic part, which correspond to the first term and the second term, respectively. The first term of the optimal control function (35) also appears in the linear-quadratic (LQ) problem of deterministic control, and Equation (38) is called the Riccati equation [14,40]. In contrast, the second term of the optimal control function (35) is new in the LQG problem of ML-DSC, and Equation (39) is called the decentralized Riccati equation in this paper.

5.3. Comparison with Completely Observable Stochastic Control or Memory-Limited Partially Observable Stochastic Control

In the LQG problem, the optimal control function of COSC is given as follows [14,40]:

\[ u^\star_i(t,s) = -R_{ii}^{-1}B_i^\top(\Psi \mu + \Psi^\top \hat{s}), \]

where \( \Psi(t) \) is the solution of the Riccati Equation (38). Function \( \Psi(t) \) appears in the second term and the first term in COSC. In addition, the optimal control function of ML-POSC is given as follows [23,24]:

\[ u^\star_i(t,z) = -R_{ii}^{-1}B_i^\top(\Psi \mu + \Pi K \hat{s}), \]

where \( \Pi(t) \) is the solution of the following partially observable Riccati equation:

\[-\frac{d\Pi}{dt} = Q + A^\top \Pi + \Pi A - \Pi BR^{-1}B^\top \Pi + (I - K)^\top \Pi BR^{-1}B^\top \Pi(I - K), \]

where \( \Pi(T) = P \) and \( K(t) \) is defined by

\[ K(t) := \left( \begin{array}{c} \Pi(t) \sum_{xz}(t) \sum_{xz}^{-1}(t) \end{array} \right). \]

Thus, the decentralized Riccati Equation (39) is a natural extension of the partially observable Riccati Equation (42) from a centralized problem to a decentralized problem.

While the first deterministic term of the optimal control function is the same for COSC (40), ML-POSC (41), and ML-DSC (35), the second stochastic term is different, reflecting the fact that the observable part of the stochastic extended state is different.

5.4. Decentralized Riccati Equation

In this subsection, we analyze the decentralized Riccati Equation (39) by comparing it with the Riccati Equation (38). The Riccati Equation (38) is the control gain matrix of deterministic control and COSC. While the Riccati Equation (38) optimizes control, it does not improve estimation, because estimation is not needed in deterministic control and COSC. In contrast, the decentralized Riccati Equation (39) may improve estimation as well as control because the controllers of ML-DSC need to estimate the state of the system and the memories of the other controllers from their own memories.

In order to support this discussion, we analyze the last term of the decentralized Riccati Equation (39), which is denoted as follows:

\[ Q_i := (I - K_i)^\top \Phi B_i R_{ii}^{-1}B_i^\top \Phi(I - K_i). \]

This term is the main difference between the Riccati Equation (38) and the decentralized Riccati Equation (39), and thus accounts for the contribution of estimation in ML-DSC. For the sake of simplicity, we focused on \( Q_N \). A similar discussion is possible for \( Q_i \) by the permutation of the controllers’ indices. We also denoted \( a := s^{-N} \) and \( b := z^N \) for notational simplicity. \( a \) is unobservable and \( b \) is observable for the controller \( N \). \( Q_N \) can be calculated as follows:

\[ Q_N = \left( \begin{array}{c} P_{aa} - P_{ab} \Sigma_{bh}^{-1} P_{ab} \Sigma_{ba} - P_{aa} \Sigma_{bb} \Sigma_{bh}^{-1} \\ -P_{ba} \Sigma_{bb}^{-1} P_{ab} \Sigma_{ba} \Sigma_{bb} P_{aa} \Sigma_{bh}^{-1} \end{array} \right), \]

for the sake of simplicity, we focused on \( Q_N \). A similar discussion is possible for \( Q_i \) by the permutation of the controllers’ indices. We also denoted \( a := s^{-N} \) and \( b := z^N \) for notational simplicity. \( a \) is unobservable and \( b \) is observable for the controller \( N \). \( Q_N \) can be calculated as follows:

\[ Q_N = \left( \begin{array}{c} P_{aa} - P_{ab} \Sigma_{bh}^{-1} P_{ab} \Sigma_{ba} - P_{aa} \Sigma_{bb} \Sigma_{bh}^{-1} \\ -P_{ba} \Sigma_{bb}^{-1} P_{ab} \Sigma_{ba} \Sigma_{bb} P_{aa} \Sigma_{bh}^{-1} \end{array} \right), \]
where $\mathcal{P}_{aa} := (\Phi B N R_{NN}^{-1} B_1^\top \Phi)_{aa}$. Because $\mathcal{P}_{aa} \succeq 0$ and $\Sigma_{bb}^{-1} \mathcal{P}_{aa} \Sigma_{ab} \Sigma_{bb}^{-1} \succeq 0$, $\Phi_{aa}$ and $\Phi_{bb}$ may be larger than $\Psi_{aa}$ and $\Psi_{bb}$, respectively. Because $\Phi_{aa}$ and $\Phi_{bb}$ are the negative feedback gains of $a$ and $b$, respectively, $\mathcal{Q}_N$ may contribute to decreasing $\Sigma_{aa}$ and $\Sigma_{bb}$. Moreover, when $\Sigma_{ab}$ is positive/negative, $\Phi_{ab}$ may be smaller/larger than $\Psi_{ab}$, which may increase/decrease $\Sigma_{ab}$. The similar discussion is possible for $\Sigma_{ba}$, $\Phi_{ba}$, and $\Psi_{ba}$ because $\Sigma$, $\Phi$, and $\Psi$ are symmetric matrices. As a result, $\mathcal{Q}_N$ may contribute to decreasing the following conditional covariance matrix:

$$
\Sigma_{a|b} := \Sigma_{aa} - \Sigma_{ab} \Sigma_{bb}^{-1} \Sigma_{ba},
$$

which corresponds to the estimation error of the unobservable $a$ from the observable $b$. It indicates that the decentralized Riccati Equation (39) may improve estimation as well as control.

It should be noted that estimation and control are not clearly separated in the LQG problem of ML-DSC. In the LQG problem of POSC, estimation and control are clearly separated, and they are optimized by the Kalman filter and the Riccati equation, respectively [11,14]. By contrast, in the LQG problem of ML-DSC, both estimation and control are optimized by the decentralized Riccati equation. This coupling of estimation and control also appears in the conventional DSC [17–22] and ML-POSC [23,24], which indicates that it may be caused by a decentralized structure and memory limitation.

6. Numerical Experiments

In this section, we demonstrate the significance of the decentralized Riccati Equation (39) by conducting the numerical experiments of two LQG problems in ML-DSC. One is the one-dimensional state case, and the other is the two-dimensional state case.

6.1. One-Dimensional State Case

In this subsection, we conduct a numerical experiment in one-dimensional state case (Figure 3a). In this case, we consider the state $x_t \in \mathbb{R}$, the observation $y^i_t \in \mathbb{R}$ and the memory $z^i_t \in \mathbb{R}$ of the controller $i \in \{1, 2\}$, which evolve by the following SDEs:

$$
\begin{align}
    dx_t &= \left( x_t + u_{1,t}^i + u_{2,t}^i \right) dt + d\omega_t, \quad (47) \\
    dy_t^i &= \left( x_t + u_{2,t}^i \right) dt + dv_t^i, \quad (48) \\
    dy_t^2 &= \left( x_t + u_{1,t}^i \right) dt + dv_t^2, \quad (49) \\
    dz_t^1 &= v_t^1 dt + dv_t^1, \quad (50) \\
    dz_t^2 &= v_t^2 dt + dv_t^2. \quad (51)
\end{align}
$$

The initial conditions are given by the standard Gaussian distributions. $\omega_t \in \mathbb{R}$, $v_t^1 \in \mathbb{R}$, and $v_t^2 \in \mathbb{R}$ are the independent standard Wiener processes. $u_{1}^i := (u_{1,t}^i, u_{2,t}^i) = \nu^i(t, z_t^i) \in \mathbb{R}^2$ and $v_t^i := \nu^i(t, z_t^i) \in \mathbb{R}$ are the controls of controller $i$. Each controller can control the memory of the other controller through $u_{y,t}^i$, which can be interpreted as the communication among the controllers. The objective function to be minimized is given as follows:

$$
J[u, v] := \mathbb{E}_{p(x_0:T, y_0:T, z_0:T, \omega, v)} \left[ \int_0^T \left( x_t^2 + \frac{2}{z_t^1} (u_{1,t}^i)^2 + (u_{2,t}^i)^2 + (v_t^i)^2 \right) dt \right], \quad (52)
$$

where $u := (u_1^1, u_2^2)$ and $v := (v_1^1, v_2^2)$. Therefore, the objective of this problem is to minimize the state variance by small controls (Figure 3b).

In this problem, the information of the controllers is neither independent nor partially nested. The information of the controller 1’s memory $z_t^1$ propagates to the controller 2’s memory $z_t^2$ through the state control $u_{1,t}^i$ and the observation control $u_{y,t}^i$ and vice versa.
While such a problem cannot be solved by the conventional DSC, ML-DSC can address it.

\[
\begin{align*}
\text{(a)} & \quad x_t \\
\text{(b)} & \quad \begin{array}{c}
\begin{array}{c}
\uparrow u^1_{y,t} \\
\uparrow u^2_{y,t}
\end{array}
\end{array} \\
\begin{array}{c}
\downarrow y^1_t \\
\downarrow y^2_t
\end{array} \\
\begin{array}{c}
\begin{array}{c}
\uparrow u^1_{x,t} \\
\uparrow u^2_{x,t}
\end{array}
\end{array} \\
\begin{array}{c}
\downarrow y^1_t \\
\downarrow y^2_t
\end{array}
\end{align*}
\]

**Figure 3.** Schematic diagram of the LQG problem in Section 6.1. (a) The state of the system \( x_t \) is one-dimensional. (b) The two controllers control the state of the system to be close to 0.

In the representation using the extended state \( s_t := (x_t, z^1_t, z^2_t) \in \mathbb{R}^3 \), the extended control \( \tilde{u}_t := (u^1_{x,t}, u^2_{x,t}, v^1_t, v^2_t) \in \mathbb{R}^4 \), and the extended standard Wiener process \( \tilde{\omega}_t := (\omega_t, v^1_t, v^2_t) \in \mathbb{R}^3 \) as in Equations (27) and (32), the SDEs defined by Equations (47)–(51) can be described as follows:

\[
ds_t = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} s_t + \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tilde{u}_t + \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tilde{u}_t^\top dt + d\tilde{\omega}_t, \tag{53}
\]

which corresponds to Equation (27). The objective function (52) can be rewritten as follows:

\[
\begin{aligned}
J[\tilde{u}] := & \mathbb{E}_{p(s_0, T; \tilde{u})} \left[ \int_0^T \left( s_t^\top \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} s_t + \sum_{i=1}^2 (\tilde{u}_t^i)^\top \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tilde{u}_t^i \right) dt \right], \tag{54}
\end{aligned}
\]

which corresponds to Equation (32). In addition, it satisfies the block diagonal matrix assumption of \( R(t) \) (31).

The Riccati Equation (38) can be solved backwards in time from the terminal condition. In contrast, the decentralized Riccati Equation (39) cannot be solved in a similar way to the Riccati Equation (38) because it depends on the covariance matrix \( \Sigma(t) \) via the estimation gain matrix \( K_i(t) \), which is the solution of the time-forward ODE (37). In order to solve the decentralized Riccati Equation (39), which is the time-backward ODE of \( \Phi(t) \), we use the forward-backward sweep method, which computes the time-forward ODE of \( \Sigma(t) \) (37) and the time-backward ODE of \( \Phi(t) \) (39) alternately [24]. We note that the partially observable Riccati Equation (42) can also be solved by the forward-backward sweep method [24].

Figure 4 shows the trajectories of \( \Psi(t), \Pi(t), \) and \( \Phi(t) \), which are the optimal control gain matrices of COSC, ML-POSC, and ML-DSC, respectively. Functions \( \Psi_{ab}(t), \Pi_{ab}(t), \) and \( \Phi_{ab}(t) \) are the negative control gains from \( b \) to \( a \). We noted that \( \Psi_{ab}(t), \Pi_{ab}(t), \) and \( \Phi_{ab}(t) \) are also the negative control gains from \( a \) to \( b \) because \( \Psi(t), \Pi(t), \) and \( \Phi(t) \) are symmetric matrices. While the elements of \( \Psi(t) \) related with the memories \( z_1 \) and \( z_2 \) are always 0, those of \( \Pi(t) \) and \( \Phi(t) \) are not (Figure 4b–f). Thus, the controls of the memories do not appear in COSC, but appear in ML-POSC and ML-DSC. This result indicates that the controls of the memories play an important role in estimation.
We first compares \( \Phi \) with \( \Psi \) in more detail to investigate \( \Phi \). \( \Phi_{xx} \) and \( \Phi_{z^i} \) are larger than \( \Psi_{xx} \) and \( \Psi_{z^i} \) (Figure 4a,d,f), which may decrease \( \Sigma_{xx} \) and \( \Sigma_{z^i} \). Moreover, \( \Phi_{x^i} \) is smaller than \( \Psi_{x^i} \) (Figure 4b,c), which may strengthen the positive correlation between \( x \) and \( z^i \). Therefore, \( \Phi_{xx} \), \( \Phi_{z^i} \), and \( \Phi_{x^i} \) may improve estimation, which is consistent with our discussion in Section 5.4. However, \( \Phi_{z^i} \) is larger than \( \Psi_{z^i} \) (Figure 4e), which may weaken the positive correlation between \( z^i \) and \( z^2 \). It seems to be contrary to our discussion because it may worsen estimation.

In order to understand \( \Phi_{z^i} \), we also compared \( \Phi \) with \( \Pi \). Estimation in ML-DSC is more challenging than that in ML-POSC because the controllers cannot completely share their information. Thus, except for \( \Phi_{z^i} \), the absolute values of \( \Phi \) are larger than those of \( \Pi \) (Figure 4a–d,f) for the same reason as the comparison with \( \Psi \). The problem is only \( \Phi_{z^i} \) (Figure 4e). In ML-POSC, the estimation between \( z^1 \) and \( z^2 \) is not needed because the controllers share their information. As a result, \( \Pi_{z^i} \) is determined only from a control perspective, not an estimation perspective. \( \Pi_{z^i} \) is almost the same with \( \Pi_{z^i} \) (Figure 4d–f), presumably because the cooperative control by the controllers 1 and 2 is more efficient than the independent control. By contrast, in ML-DSC, the estimation between \( z^1 \) and \( z^2 \) is necessary because each controller cannot directly access the other controller. \( \Phi_{z^i} \) is smaller than \( \Pi_{z^i} \) (Figure 4e), which may strengthen the positive correlation between \( z^1 \) and \( z^2 \). Therefore, \( \Phi_{z^i} \) may be determined by a trade-off between control and estimation.

\[
\begin{align*}
\Phi_{z^i} &= -B_i^{-1} (\Psi \mu + \Psi K \hat{s}), \\
\Pi_{z^i} &= -B_i^{-1} (\Psi \mu + \Pi K \hat{s}),
\end{align*}
\]

which replaced \( \Phi \) with \( \Psi \) and \( \Pi \), respectively. We noted that the first terms are not important because \( \mu(t) = 0 \) is satisfied in this set-up. The result is shown in Figure 5. The variances of the state and the memories are \( u^* > u^\Pi > u^\Psi \) (Figure 5a–c). Similarly, the expected cumulative costs are \( u^\Psi > u^\Pi > u^* \) (Figure 5d). These orders are consistent with the extent
of the optimization of estimation. \( u^\Psi \) does not take into account estimation at all, and its performance is the worst. \( u^{\Pi} \) takes into account only the state estimation, and it performs better than \( u^\Psi \), but not optimally. \( u^* \) takes into account the estimation of the other memories and the state, and its performance is optimal.

Figure 5. Stochastic simulations in Section 6.1. (a–c) Stochastic behaviors of (a) the state \( x_t \), (b) the controller 1’s memory \( z^1_t \), and (c) the controller 2’s memory \( z^2_t \) for 100 samples in Section 6.1. (d) The expected cumulative cost computed from the 100 samples. Blue, orange, and green curves are controlled by \( u^\Psi \) (55), \( u^{\Pi} \) (56), and \( u^* \) (35), respectively.

6.2. Two-Dimensional State Case

In this subsection, we conduct a numerical experiment in two-dimensional state case (Figure 6a). In this case, we formulated the target state \( x^\text{tar}_t := (x^\text{tar,1}_t, x^\text{tar,2}_t) \in \mathbb{R}^2 \), the actual state \( x^\text{act}_t := (x^\text{act,1}_t, x^\text{act,2}_t) \in \mathbb{R}^2 \), the observation \( y_t := (y^1_t, y^2_t) \in \mathbb{R}^2 \), and the memory \( z_t := (z^1_t, z^2_t) \in \mathbb{R}^2 \) as follows (Figure 6b):

\[
\begin{align*}
    dx^\text{tar,1}_t &= -2\pi x^\text{tar,2}_t \, dt, \\
    dx^\text{tar,2}_t &= 2\pi x^\text{tar,1}_t \, dt, \\
    dx^\text{act,1}_t &= \left(-2\pi x^\text{act,2}_t + u^1_t\right) \, dt + d\omega^1_t, \\
    dx^\text{act,2}_t &= \left(2\pi x^\text{act,1}_t + u^2_t\right) \, dt + d\omega^2_t, \\
    dy^1_t &= \left(x^\text{act,1}_t - x^\text{tar,1}_t + u^2_y\right) \, dt + dv^1_t, \\
    dy^2_t &= \left(x^\text{act,2}_t - x^\text{tar,2}_t + u^1_y\right) \, dt + dv^2_t, \\
    dz^1_t &= v^1_t \, dt + dy^1_t, \\
    dz^2_t &= v^2_t \, dt + dy^2_t,
\end{align*}
\]
where $x_0^{br} = (10, 0), x_0^{act} \sim \mathcal{N}(x_0^{act}|(10, 0), I), y_0 \sim \mathcal{N}(y_0|0, I)$, and $z_0 \sim \mathcal{N}(z_0|0, I). \omega^i_t \in \mathbb{R}$ and $v^i_t \in \mathbb{R}$ are the independent standard Wiener processes. $u^i_t := (u^i_{x,t}, u^i_{y,t}) = u^i(t, z^i_t) \in \mathbb{R}^2$ and $v^i_t := v^i(t, z^i_t) \in \mathbb{R}$ are the controls of controller $i$. The objective function to be minimized is given as follows:

$$J[u, v] := \mathbb{E}_{u,v} \left[ \int_0^T \sum_{i=1}^2 \left( 10 \left( x^i_{t} - x^i_{tar} \right)^2 + (u^i_t)^2 + (v^i_t)^2 \right) dt \right]. \quad (65)$$

The objective of this problem is to minimize the distance of the actual state $x^i_{t}$ from the rotating target state $x^i_{tar}$ by small controls. The solution of Equations (57) and (58) is given by $x^i_{tar} = (x^i_{tar,1}, x^i_{tar,2}) = (10 \cos(2\pi t), 10 \sin(2\pi t))$. If Equations (59) and (60) do not have the standard Wiener processes $d\omega^i_t$ and $d\omega^j_t$, respectively, $u^i_{t,1} = 0$ and $u^j_{t,2} = 0$ are optimal because $x^i_{tar} = x^j_{tar}$ is satisfied. In practice, however, the actual state $x^i_{t}$ does not coincide with the target state $x^i_{tar}$ without the control $u^i_t$ due to the state noise $d\omega^i_t$, and needs to be controlled. Controller $i$ observes and controls the actual state $x^i_{t}$ only $x^i_{t,1}$-axis direction. As a result, the communication between the controllers is more important in this problem.

![Figure 6](https://example.com/figure6.png)

**Figure 6.** Schematic diagram of the LQG problem in Section 6.2. (a) The state of the system $x_t = (x^1_t, x^2_t)$ is two-dimensional. (b) The two controllers control the actual state $x^i_{t,1}$ to be close to the target state $x^i_{t,1} = (10 \cos(2\pi t), 10 \sin(2\pi t))$. Controller $i$ observes and controls the actual state $x^i_{t,1}$ only $x^i_{t,1}$-axis direction.

By defining the state $x_t := x^{act}_t - x^{tar}_t$, Equations (57)–(64) are converted as follows:

$$dx^1_t = (-2\pi x^1_t + u^1_{x,t}) dt + d\omega^1_t, \quad (66)$$

$$dx^2_t = (2\pi x^2_t + u^2_{x,t}) dt + d\omega^2_t, \quad (67)$$

$$dy^1_t = \left( x^1_t + u^1_{y,t} \right) dt + dv^1_t, \quad (68)$$

$$dy^2_t = \left( x^2_t + u^2_{y,t} \right) dt + dv^2_t, \quad (69)$$

$$dz^1_t = v^1_t dt + dy^1_t, \quad (70)$$

$$dz^2_t = v^2_t dt + dy^2_t, \quad (71)$$
where the initial conditions are given by the standard Gaussian distributions. Furthermore, the objective function (65) is converted as follows:

\[
J[u, v] := \mathbb{E}_{u, v} \left[ \int_{0}^{10} \sum_{i=1}^{2} \left( 10(x_i^t)^2 + (u_i^t)^2 + (v_i^t)^2 \right) dt \right].
\]  

(72)

As a result, the problem of controlling the actual state \(x^\text{act}_1\) to be close to the target state \(x^\text{tar}_1\) is equivalent to the problem of controlling the state \(x_1\) to be close to 0.

In the representation using the extended state \(s_t := (x_1^t, x_2^t, z_1^t, z_2^t) \in \mathbb{R}^4\), the extended control \(\tilde{u}_i^t := (u_{x_1}^t, u_{v_2}^t, v_1^t) = \tilde{u}^t(t, z_1^t) \in \mathbb{R}^3\), and the extended standard Wiener process \(\tilde{\omega}_i := (\omega_1^t, \omega_2^t, v_1^t, v_2^t) \in \mathbb{R}^4\) as in Equations (27) and (32), the SDEs defined by Equations (66)–(71) can be described as follows:

\[
ds_t = \begin{pmatrix} 0 & -2\pi & 0 & 0 \\ 2\pi & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} s_t + \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tilde{u}_i^t + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tilde{\omega}_i^t dt + d\tilde{\omega}_i^t,
\]  

(73)

which corresponds to Equation (27). The objective function (72) can be rewritten as follows:

\[
J[\tilde{u}] := \mathbb{E}_{\tilde{u}, s_0, \tilde{\omega}} \left[ \int_{0}^{10} \begin{pmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} s_t + \sum_{i=1}^{2} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \tilde{u}_i^t \right] dt,
\]  

(74)

which corresponds to Equation (32). In addition, it satisfies the block diagonal matrix assumption of \(R(t)\) (31).

Figure 7 shows the trajectories of \(\Psi(t), \Pi(t),\) and \(\Phi(t)\). Unlike \(\Psi(t)\), the elements of \(\Pi(t)\) and \(\Phi(t)\) related with the memories \(z_1\) and \(z_2\) are not always 0 (Figure 7e–j), which indicates that the controls of the memories appear in ML-POSC and ML-DSC. The elements of \(\Phi(t)\) only largely deviate from those of \(\Pi(t)\) in Figure 7g–i. Figure 7g,h show the negative feedback control gain of the memory \(z^1\). \(\Phi_{z^1}^\text{act}(t)\) is larger than \(\Pi_{z^1}^\text{act}(t)\), which indicates that the memories in ML-DSC are controlled more strongly than those in ML-POSC. Figure 7i shows the control gain between the state \(x_1^1\) and the memory \(z_2^1\). While the controller 1 can control the state \(x_1^1\) based on the memory \(z_1^1\) in ML-POSC, it cannot in ML-DSC, because the controllers do not share their memories in ML-DSC. Furthermore, while controller 1 does not need to send the information of the state \(x_1^1\) to controller 2’s memory \(z_2^1\), this is required in ML-DSC. As a result, \(\Phi_{x_1^1 z_2^1}(t)\) differs greatly from \(\Pi_{x_1^1 z_2^1}(t)\). A similar discussion is possible for Figure 7j.

In order to clarify the significance of the decentralized Riccati Equation (39), we compared the performance of the optimal control function \(u^*\) (35) with that of the control functions \(u^\Psi\) (55) and \(u^\Pi\) (56). The result is shown in Figure 8. The actual state \(x^\text{act}\) faithfully tracks the target state \(x^\text{tar}\) under the optimal control function \(u^*\) (Figure 8a–c (green)). Similarly, the memory \(z\) is stably controlled in the optimal control function \(u^*\) (Figure 8d,e (green)). As a result, the performance of the optimal control function \(u^*\) is optimal (Figure 8f (green)).
Figure 7. (a–j) Trajectories of the elements of $\Psi(t) \in \mathbb{R}^{4 \times 4}$ (blue), $\Pi(t) \in \mathbb{R}^{4 \times 4}$ (orange), and $\Phi(t) \in \mathbb{R}^{4 \times 4}$ (green) in Section 6.2. They are the solutions of the Riccati Equation (38), the partially observable Riccati Equation (42), and the decentralized Riccati Equation (39), respectively. $\Psi(t)$, $\Pi(t)$, and $\Phi(t)$ are symmetric matrices, and duplicate elements are not visualized.

Figure 8. Stochastic simulations in Section 6.2. (a–e) Stochastic behaviors of the actual state $x_{\text{act}}^t = (x_{\text{act},1}^t, x_{\text{act},2}^t)$ (a–c) and the memory $z_t = (z_1^t, z_2^t)$ (d,e) for 100 samples. (f) The expected cumulative cost computed from the 100 samples. Black curves indicate the target state $x_{\text{tar}}^t$. Blue, orange, and green curves are controlled by $u^\Psi$ (55), $u^{\Pi}$ (56), and $u^*$ (35), respectively.

7. Discussion

In this paper, we proposed ML-DSC, which explicitly formulates the finite-dimensional memories of the controllers. In ML-DSC, each controller is designed to compress the infinite-dimensional observation histories appropriately into the finite-dimensional memory, and determines the optimal control based on it. As a result, ML-DSC can handle the difficulty in the conventional DSC that arises from the finiteness of actual memory of controllers. We demonstrated the effectiveness of ML-DSC in the LQG problem. While the conventional
DSC needs to restrict the interaction among the controllers to solve the LQG problem, ML-DSC is free from such a restriction. We found that estimation and control are optimized by the decentralized Riccati equation in the LQG problem of ML-DSC. Our numerical experiments showed that the decentralized Riccati equation is superior to the Riccati equation and the partially observable Riccati equation in this problem.

ML-DSC can also address non-LQG problems. In DSC, the non-LQG problem cannot be solved numerically even if the number of the controllers is one, which corresponds to POSC [12,13]. This is because a functional differential equation needs to be solved in the non-LQG problem of POSC, which is generally intractable, even numerically. ML-POSC and ML-DSC are more tractable than the conventional POSC and DSC because only HJB-FP equations need to be solved, which are partial differential equations. The previous work showed that ML-POSC is more effective than the conventional POSC in a non-LQG problem [23]. Therefore, unlike the conventional DSC, ML-DSC may also be effective to non-LQG problems.

In order to solve ML-DSC with a large number of controllers, more efficient numerical algorithms are needed because HJB-FP equations become high-dimensional partial differential equations. In order to solve high-dimensional HJB-FP equations, neural network-based algorithms have been proposed in mean-field stochastic game and control [50,51]. Therefore, by exploiting these neural network-based algorithms, we may efficiently solve ML-DSC with a large number of controllers.

Author Contributions: Conceptualization, Formal analysis, Funding acquisition, Writing—original draft, T.T. and T.J.K.; Software, Visualization, T.T. All authors have read and agreed to the published version of the manuscript.

Funding: The first author received a JSPS Research Fellowship (Grant No. 21J20436). This work was supported by JSPS KAKENHI (Grant No. 19H05799) and JST CREST (Grant No. JPMJCR2011).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations
The following abbreviations are used in this manuscript:

| Acronym   | Description                                      |
|-----------|--------------------------------------------------|
| COSC      | Completely Observable Stochastic Control         |
| POSC      | Partially Observable Stochastic Control          |
| DSC       | Decentralized Stochastic Control                |
| ML-POSC   | Memory-Limited Partially Observable Stochastic Control |
| ML-DSC    | Memory-Limited Decentralized Stochastic Control  |
| DEC-POMDP | Decentralized Partially Observable Markov Decision Process |
| HJB       | Hamilton-Jacobi-Bellman                         |
| FP        | Fokker-Planck                                   |
| ODE       | Ordinary Differential Equation                  |
| SDE       | Stochastic Differential Equation                 |
| LQ        | Linear-Quadratic                                |
| LQG       | Linear-Quadratic-Gaussian                       |

Appendix A. Proof of Theorem 1
In this section, we prove Theorem 1, which is Pontryagin’s minimum principle on the probability density function space. In this paper, we prove Pontryagin’s minimum principle via Bellman’s dynamic programming principle, which is a similar approach with reference [23]. We note that Pontryagin’s minimum principle can also be proved directly, which is almost the same with reference [24] and omitted in this paper.
Appendix A.1. Bellman’s Dynamic Programming Principle

In this subsection, we obtain the optimality condition of ML-DSC from the viewpoint of Bellman’s dynamic programming principle on the probability density function space. The minimization of the objective function can be calculated as follows:

\[
\min_u f[u] = \min_{u_0, T} \left[ \int_0^T \tilde{f}(\tau, p_\tau, u_\tau) d\tau + \tilde{g}(p_T) \right] \\
= \min_{u_0, T} \left[ \int_0^{t-dt} \tilde{f}(\tau, p_\tau, u_\tau) d\tau + \int_{t+dt}^T \tilde{f}(\tau, p_\tau, u_\tau) d\tau + \tilde{g}(p_T) \right].
\]  

(A1)

Therefore, the optimal joint control function at time \( t \) is given as follows:

\[
u_t^* = \arg \min_{u_t} \left[ f(t, p_t, u_t) d\tau + \min_{u_{t+dt}} \left[ \int_{t+dt}^T \tilde{f}(\tau, p_\tau, u_\tau) d\tau + \tilde{g}(p_T) \right] \right].
\]  

(A2)

We define the value function

\[
V(t, p) := \min_{u_t, T} \left[ \int_0^T \tilde{f}(t, p_\tau, u_\tau) d\tau + \tilde{g}(p_T) \right].
\]  

(A3)

where \( \{p_\tau | \tau \in [t, T]\} \) is the solution of FP Equation (16) with \( p_t = p \). We note that \( V(T, p) = \tilde{g}(p) \) is satisfied. From the definition of the value function, the optimal joint control function at time \( t \) can be calculated as follows:

\[
u_t^* = \arg \min_{u_t} \left[ f(t, p_t, u_t) dt + V(t+dt, p_t + L^*_u p_t dt) \right] \\
= \arg \min_{u_t} \left[ f(t, p_t, u_t) dt + V(t, p_t) + \frac{\partial V(t, p_t)}{\partial t} dt \right. \\
+ \left. \left( \int \left( \frac{\delta V(t, p_t)}{\delta p} \right)(s_t) L^*_u p_t(s_t) ds_t \right) dt \right] \\
= \arg \min_{u_t} \left[ f(t, p_t, u_t) + \int \left( \frac{\delta V(t, p_t)}{\delta p} \right)(s_t) L^*_u p_t(s_t) ds_t \right].
\]  

(A4)

Since \( L^*_u \) is the conjugate of \( L_u \) (22),

\[
u_t^* = \arg \min_{u_t} \left[ f(t, p_t, u_t) + \int p_t(s_t) L_u \left( \frac{\delta V(t, p_t)}{\delta p} \right)(s_t) ds_t \right].
\]  

(A5)

From the definition of the Hamiltonian (20),

\[
u_t^* = \arg \min_{u_t} \mathbb{E}_{p_t(s_t)} \left[ H(t, s_t, u_t, \frac{\delta V(t, p_t)}{\delta p})(s_t) \right].
\]  

(A6)

The minimization of the Hamiltonian can be calculated as follows:

\[
\min_{u_t} \mathbb{E}_{p_t(s_t)} \left[ H(t, s_t, u_t, \frac{\delta V(t, p_t)}{\delta p})(s_t) \right] \\
= \min_{u_t'} \mathbb{E}_{p_t(s_t)} \left[ H(t, s_t, u_t' - i s, \frac{\delta V(t, p_t)}{\delta p})(s_t) \right].
\]  

(A7)
Since $u^*_i$ is the function of $z_i^t$ in ML-DSC, the minimization by $u^*_i$ can be exchanged with the expectation by $p(z_i^t)$ as follows:

$$
\min_{u^*_i} \mathbb{E}_{p(z_i^t)} \left[ \mathcal{H} \left( t, s_t, (u^*_i, u_t), \frac{\delta V(t, p_t)}{\delta p} (s_t) \right) \right]
$$

$$
= \mathbb{E}_{p(z_i^t)} \left[ \min_{u^*_i} \mathbb{E}_{p(z_i^t)} \left[ \mathcal{H} \left( t, s_t, (u^*_i, u_t), \frac{\delta V(t, p_t)}{\delta p} (s_t) \right) \right] \right]. \quad (A8)
$$

Therefore, the optimal control function of controller $i$ at time $t$ is given as follows:

$$
u^*_i(z_i^t) = \arg \min_{u^*_i} \mathbb{E}_{p(z_i^t)} \left[ \mathcal{H} \left( t, s_t, (u^*_i, u_t), \frac{\delta V(t, p_t)}{\delta p} (s_t) \right) \right]. \quad (A9)
$$

In order to obtain the optimal control function (A9), we need to obtain the value function $V(t, p)$. The value function $V(t, p)$ can be calculated as follows:

$$
V(t, p) = \min_u \left[ \int_t^T \tilde{f}(\tau, p, u, u^*_t) d\tau + \tilde{g}(p_T) \right]
$$

$$
= \min_u \left[ \tilde{f}(t, p, u) dt + \min_u \left[ \int_t^{T+dt} \tilde{f}(\tau, p, u) d\tau + \tilde{g}(p_T) \right] \right]
$$

$$
= \min_u \left[ \tilde{f}(t, p, u) dt + V(t + dt, p + \mathcal{L}_u^t p dt) \right]
$$

$$
= \min_u \left[ \tilde{f}(t, p, u) dt + V(t, p) + \frac{\partial V(t, p)}{\partial t} dt + \left( \int \frac{\delta V(t, p)}{\partial p} (s) \mathcal{L}_u^t p(s) ds \right) dt \right]. \quad (A10)
$$

Therefore, the following equation is obtained:

$$
- \frac{\partial V(t, p)}{\partial t} = \min_u \left[ \tilde{f}(t, p, u) + \int \frac{\delta V(t, p)}{\partial p} (s) \mathcal{L}_u^t p(s) ds \right]. \quad (A11)
$$

Since $\mathcal{L}_u^t$ is the conjugate of $\mathcal{L}_u$ (22),

$$
- \frac{\partial V(t, p)}{\partial t} = \min_u \left[ \tilde{f}(t, p, u) + \int p(s) \mathcal{L}_u (t) \frac{\delta V(t, p)}{\delta p} (s) ds \right]
$$

$$
= \min_u \mathbb{E}_{p(s)} \left[ \tilde{f}(t, s, u) + \mathcal{L}_u (t) \frac{\delta V(t, p)}{\delta p} (s) \right] ds. \quad (A12)
$$

From the definition of the Hamiltonian $\mathcal{H}$ (20),

$$
- \frac{\partial V(t, p)}{\partial t} = \min_u \mathbb{E}_{p(s)} \left[ \mathcal{H} \left( t, s, u, \frac{\delta V(t, p)}{\delta p} (s) \right) \right]. \quad (A13)
$$

From Equation (A6),

$$
- \frac{\partial V(T, p)}{\partial t} = \mathbb{E}_{p(s)} \left[ \mathcal{H} \left( t, s, u^*_t, \frac{\delta V(t, p)}{\delta p} (s) \right) \right], \quad (A14)
$$

where $V(T, p) = \tilde{g}(p)$. This functional differential Equation (A14) is called Bellman equation. Therefore, from Bellman’s dynamic programming principle on the probability density function space, the optimal control function of ML-DSC (A9) is obtained by solving FP Equation (16) and Bellman Equation (A14).

However, Bellman Equation (A14) is a functional differential equation, which cannot be solved even numerically. Therefore, Bellman’s dynamic programming principle on the probability density function space is not practical. In the next subsection, we resolve this problem by converting Bellman’s dynamic programming principle to Pontryagin’s
minimum principle on the probability density function space. This conversion technique has also been used in mean-field stochastic control [25,26] and ML-POSC [23,24].

Appendix A.2. Conversion from Bellman’s Dynamic Programming Principle to Pontryagin’s Minimum Principle

In this subsection, we prove Theorem 1 by converting Bellman’s dynamic programming principle to Pontryagin’s minimum principle on the probability density function space.

We first define

\[ W(t, p, s) := \frac{\delta V(t, p)}{\delta p}(s), \]  

(A15)

which satisfies \( W(T, p, s) = g(s) \). Differentiating Bellman Equation (A14) with respect to \( p \), the following equation is obtained:

\[ -\frac{\partial W(t, p, s)}{\partial t} = \mathcal{H}(t, s, u^*, W) + \mathbb{E}_{p(u')} \left[ \mathcal{L}_{u^*} \frac{\delta W(t, p, s')}{\delta p}(s) \right]. \]  

(A16)

Since \( \mathcal{L}_{u^*} \) is the conjugate of \( \mathcal{L}_{u^*}^+ \) (22),

\[ -\frac{\partial W(t, p, s)}{\partial t} = \mathcal{H}(t, s, u^*, W) + \int \frac{\delta W(t, p, s)}{\delta p}(s') \mathcal{L}_{u^*}^+ p(s') ds'. \]  

(A17)

We then define

\[ \omega(t, s) := W(t, p_t, s), \]  

(A18)

where \( p_t \) is the solution of FP Equation (16). The time derivative of \( \omega(t, s) \) can be calculated as follows:

\[ \frac{\partial \omega(t, s)}{\partial t} = \frac{\partial W(t, p_t, s)}{\partial t} + \int \frac{\delta W(t, p_t, s)}{\delta p}(s') \frac{\partial p_t(s')}{\partial t} ds'. \]  

(A19)

By substituting Equation (A17) into Equation (A19), the following equation is obtained:

\[ -\frac{\partial \omega(t, s)}{\partial t} = \mathcal{H}(t, s, u^*, \omega) - \int \frac{\delta W(t, p_t, s)}{\delta p}(s') \left( \frac{\partial p_t(s')}{\partial t} - \mathcal{L}_{u^*} p_t(s') \right) ds'. \]  

(A20)

From FP Equation (16), \((*) = 0\) holds. Therefore, HJB Equation (24) is obtained.

From Equations (A15) and (A18), the optimal control function of ML-DSC (A9) can be calculated as follows:

\[ u^{i*}(t, z') = \arg \min_{u^i} \mathbb{E}_{p_i(s_{i-1}|s')} \left[ \mathcal{H} \left( t, s, (u^{i*}, u), \omega \right) \right]. \]  

(A21)

Therefore, the optimal control function of ML-DSC (A9) is obtained by solving FP Equation (16) and HJB Equation (24).

Appendix B. Proof of Theorem 2

In the LQG problem, the Hamiltonian (20) is given as follows:

\[ \mathcal{H}(t, s, u, w) = s^T Q s + \sum_{i=1}^{N} (u_i^*)^T R_i u_i + \left( \frac{\partial \omega(t, s)}{\partial s} \right)^T \left( A s + \sum_{i=1}^{N} B_i u^i \right) + \frac{1}{2} \text{tr} \left\{ \frac{\partial}{\partial s} \left( \frac{\partial \omega(t, s)}{\partial s} \right)^T \sigma \sigma^T \right\}. \]  

(A22)
Therefore, in the LQG problem, the optimal control function of ML-DSC (19) can be calculated as follows:

\[ u^{is}(t, z^i) = -\frac{1}{2} R_{ii}^{-1} B_i^\top \mathbb{E}_{\mu(s^i|z^i)} \left[ \frac{\partial u(t, s)}{\partial s} \right]. \] (A23)

We assume that \( p_i(s) \) is the Gaussian distribution

\[ p_i(s) := \mathcal{N}(s|\mu(t), \Sigma(t)), \] (A24)

and \( w(t, s) \) is the quadratic function

\[ w(t, s) = s^\top \Phi(t)s + \alpha^\top(t)s + \beta(t). \] (A25)

In this case, the optimal control function of ML-DSC (A23) can be calculated as follows:

\[ u^{is}(t, z^i) = -\frac{1}{2} R_{ii}^{-1} B_i^\top (2\Phi K_i \beta + 2\Phi \mu + \alpha). \] (A26)

Since Equation (A26) is linear with respect to \( \hat{s} \), \( p_i(s) \) becomes the Gaussian distribution, which is consistent with our assumption (A24).

Substituting Equations (A25) and (A26) into HJB Equation (24), the following ODEs are obtained:

\[ \frac{d\Phi}{dt} = Q + A^\top \Phi + \Phi A - \Phi B R^{-1} B^\top \Phi + \mathcal{Q}, \] (A27)

\[ \frac{d\alpha}{dt} = (A - BR^{-1}B^\top \Phi)^\top \alpha - 2\mathcal{Q}\mu, \] (A28)

\[ \frac{d\beta}{dt} = \text{tr} \left( \Phi \sigma \sigma^\top \right) \right) - \frac{1}{4} \alpha^\top BR^{-1} B^\top \alpha + \mu^\top \mathcal{Q}\mu, \] (A29)

where \( \mathcal{Q} := \sum_{i=1}^{N} (I - K_i)^\top \Phi_i B_i R_i^{-1} B_i^\top \Phi_i (I - K_i), \) \( \Phi(T) = P, \alpha(T) = 0, \) and \( \beta(T) = 0. \) If \( \Phi(t), \alpha(t), \) and \( \beta(t) \) satisfy ODEs (A27), (A28), and (A29), respectively, HJB Equation (24) is satisfied, which is consistent with our assumption (A25).

Defining \( Y(t) \) by \( \alpha(t) = 2Y(t)\mu(t) \) and \( \Psi(t) \) by \( \Psi(t) := \Phi(t) + Y(t), \) the optimal control function of ML-DSC (A26) can be calculated as follows:

\[ u^{is}(t, z^i) = -R_{ii}^{-1} B_i^\top (\Psi \mu + \Phi K_i \beta). \] (A30)

From Equations (A27) and (A28), \( \Psi(t) \) is the solution of the Riccati Equation (38). From Equation (A27), \( \Phi(t) \) is the solution of the decentralized Riccati Equation (39). The detailed calculations are almost the same with reference [23].

References

1. Mahajan, A.; Teneketzis, D. On the design of globally optimal communication strategies for real-time noisy communication systems with noisy feedback. *IEEE J. Sel. Areas Commun.* 2008, 26, 580–595. [CrossRef]
2. Mahajan, A.; Teneketzis, D. Optimal Design of Sequential Real-Time Communication Systems. *IEEE Trans. Inf. Theory* 2009, 55, 5317–5338. [CrossRef]
3. Nayyar, A.; Teneketzis, D. Sequential Problems in Decentralized Detection with Communication. *IEEE Trans. Inf. Theory* 2011, 57, 5410–5435. [CrossRef]
4. Mahajan, A.; Teneketzis, D. Optimal Performance of Networked Control Systems with Nonclassical Information Structures. *SIAM J. Control Optim.* 2009, 48, 1377–1404. [CrossRef]
5. Witsenhausen, H.S. A Counterexample in Stochastic Optimum Control. *SIAM J. Control* 1968, 6, 131–147. [CrossRef]
6. Nayyar, A.; Mahajan, A.; Teneketzis, D. Decentralized Stochastic Control with Partial History Sharing: A Common Information Approach. *IEEE Trans. Autom. Control* 2013, 58, 1644–1658. [CrossRef]
7. Mahajan, A.; Nayyar, A. Sufficient Statistics for Linear Control Strategies in Decentralized Systems With Partial History Sharing. *IEEE Trans. Autom. Control* 2015, 60, 2046–2056. [CrossRef]
8. Charalambous, C.D.; Ahmed, N.U. Team Optimality Conditions of Distributed Stochastic Differential Decision Systems with Decentralized Noisy Information Structures. *IEEE Trans. Autom. Control* 2017, 62, 708–723. [CrossRef]
9. Charalambous, C.D.; Ahmed, N.U. Centralized Versus Decentralized Optimization of Distributed Stochastic Differential Decision Systems with Different Information Structures—Part I: A General Theory. *IEEE Trans. Autom. Control* 2017, 62, 1194–1209. [CrossRef]

10. Charalambous, C.D.; Ahmed, N.U. Centralized Versus Decentralized Optimization of Distributed Stochastic Differential Decision Systems with Different Information Structures—Part II: Applications. *IEEE Trans. Autom. Control* 2018, 63, 1913–1928. [CrossRef]

11. Wonham, W.M. On the Separation Theorem of Stochastic Control. *SIAM J. Control* 1968, 6, 312–326. [CrossRef]

12. Bensoussan, A. *Stochastic Control of Partially Observable Systems*; Cambridge University Press: Cambridge, UK, 1992. [CrossRef]

13. Nisio, M. Stochastic Control Theory. In *Probability Theory and Stochastic Modelling*; Springer: Tokyo, Japan, 2015; Volume 72. [CrossRef]

14. Bensoussan, A. Estimation and Control of Dynamical Systems. In *Interdisciplinary Applied Mathematics*; Springer International Publishing: Cham, Switzerland, 2018; Volume 48. [CrossRef]

15. Wang, G.; Wu, Z.; Xiong, J. An Introduction to Optimal Control of FBSDE with Incomplete Information; SpringerBriefs in Mathematics; Springer International Publishing: Cham, Switzerland, 2018. [CrossRef]

16. Bensoussan, A.; Yam, S.C.P. Mean field approach to stochastic control with partial information. *ESAIM Control Optim. Calc. Var.* 2021, 27, 89. [CrossRef]

17. Lessard, L.; Lall, S. A state-space solution to the two-player decentralized optimal control problem. In Proceedings of the 2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 28–30 September 2011; pp. 1559–1564. [CrossRef]

18. Lessard, L.; Lall, S. Optimal controller synthesis for the decentralized two-player problem with output feedback. In Proceedings of the 2012 American Control Conference (ACC), Montréal, QC, Canada, 27–29 June 2012; pp. 6314–6321. [CrossRef]

19. Lessard, L. Decentralized LQG control of systems with a broadcast architecture. In Proceedings of the 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), Maui, HI, USA, 10–13 December 2012; pp. 6241–6246. [CrossRef]

20. Lessard, L.; Nayyar, A. Structural results and explicit solution for two-player LQG systems on a finite time horizon. In Proceedings of the 52nd IEEE Conference on Decision and Control, Firenze, Italy, 10–13 December 2013; pp. 6542–6549. [CrossRef]

21. Lessard, L.; Lall, S. Optimal Control of Two-Player Systems With Output Feedback. *IEEE Trans. Autom. Control* 2015, 60, 2129–2144. [CrossRef]

22. Nayyar, A.; Lessard, L. Structural results for partially nested LQG systems over graphs. In Proceedings of the 2015 American Control Conference (ACC), Chicago, IL, USA, 1–3 July 2015; pp. 5457–5464. [CrossRef]

23. Tottori, T.; Kobayashi, T.J. Memory-Limited Partially Observable Stochastic Control and Its Mean-Field Control Approach. *Entropy* 2022, 24, 1599. [CrossRef]

24. Tottori, T.; Kobayashi, T.J. Forward-Backward Sweep Method for the System of HJB-FP Equations in Memory-Limited Partially Observable Stochastic Control. *Entropy* 2023, 25, 208. [CrossRef]

25. Bensoussan, A.; Frehse, J.; Yam, S.C.P. The Master equation in mean field theory. *J. Math. Pures Appl.* 2015, 103, 1441–1474. [CrossRef]

26. Bensoussan, A.; Frehse, J.; Yam, S.C.P. On the interpretation of the Master Equation. *Stoch. Process. Their Appl.* 2017, 127, 2093–2137. [CrossRef]

27. Bensoussan, A.; Frehse, J.; Yam, P. *Mean Field Games and Mean Field Type Control Theory*; Springer Briefs in Mathematics; Springer: New York, NY, USA, 2013. [CrossRef]

28. Carmona, R.; Delarue, F. Probabilistic Theory of Mean Field Games with Applications I. In *Probability Theory and Stochastic Modelling*; Springer Nature: Cham, Switzerland, 2018; Volume 83. [CrossRef]

29. Carmona, R.; Delarue, F. Probabilistic Theory of Mean Field Games with Applications II. In *Probability Theory and Stochastic Modelling*; Springer International Publishing: Cham, Switzerland, 2018; Volume 84. [CrossRef]

30. Achdou, Y. Finite Difference Methods for Mean Field Games. In *Hamilton-Jacobi Equations: Approximations, Numerical Analysis and Applications: Cetraro, Italy 2011*; Loreti, P., Tchou, N.A., Eds.; Lecture Notes in Mathematics, Springer: Berlin/Heidelberg, Germany, 2013; pp. 1–47. [CrossRef]

31. Achdou, Y.; Laurière, M. Mean Field Games and Applications: Numerical Aspects. In *Mean Field Games: Cetraro, Italy 2017*; Achdou, Y., Cardaliaguet, P., Delarue, F., Porretta, A., Santambrogio, F., Cardaliaguet, P., Porretta, A., Eds.; Lecture Notes in Mathematics, Springer: Berlin/Heidelberg, Germany, 2017; pp. 312–326. [CrossRef]

32. Laurière, M. Numerical Methods for Mean Field Games and Mean Field Type Control. *arXiv* 2021, arXiv:2106.06231. https://doi.org/10.48550/arXiv.2106.06231.

33. Bernstein, D.S. Bounded Policy Iteration for Decentralized POMDPs. In Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, UK, 30 July–5 August 2005; pp. 1287–1292.

34. Bernstein, D.S.; Amato, C.; Hansen, E.A.; Zilberstein, S. Policy Iteration for Decentralized Control of Markov Decision Processes. *J. Artif. Intell. Res.* 2009, 34, 89–132. [CrossRef]

35. Amato, C.; Bernstein, D.S.; Zilberstein, S. Optimizing Memory-Bounded Controllers for Decentralized POMDPs. In Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, 19–22 July 2007. [CrossRef]

36. Amato, C.; Bonet, B.; Zilberstein, S. Finite-State Controllers Based on Mealy Machines for Centralized and Decentralized POMDPs. *Proc. AAAI Conf. Artif. Intell.* 2010, 24, 1052–1058. [CrossRef]
37. Kumar, A.; Zilberstein, S. Anytime Planning for Decentralized POMDPs using Expectation Maximization. In Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, 8–11 July 2010; p. 9.

38. Oliehoek, F.A.; Amato, C. A Concise Introduction to Decentralized POMDPs; SpringerBriefs in Intelligent Systems; Springer International Publishing: Cham, Switzerland, 2016. [CrossRef]

39. Tottie, T.; Kobayashi, T.J. Forward and Backward Bellman Equations Improve the Efficiency of the EM Algorithm for DEC-POMDP. *Entropy* **2021**, *23*, 551. [CrossRef]

40. Yong, J.; Zhou, X.Y. *Stochastic Controls*; Springer: New York, NY, USA, 1999. [CrossRef]

41. Kushner, H. Optimal stochastic control. *IRE Trans. Autom. Control* **1962**, *7*, 120–122. [CrossRef]

42. Carlini, E.; Silva, F.J. Semi-Lagrangian schemes for mean field game models. In Proceedings of the 52nd IEEE Conference on Decision and Control, Firenze, Italy, 10–13 December 2013; pp. 3115–3120. [CrossRef]

43. Carlini, E.; Silva, F.J. A Fully Discrete Semi-Lagrangian Scheme for a First Order Mean Field Game Problem. *SIAM J. Numer. Anal.* **2014**, *52*, 45–67. [CrossRef]

44. Carlini, E.; Silva, F.J. A semi-Lagrangian scheme for a degenerate second order mean field game system. *Discret. Contin. Dyn. Syst.* **2015**, *35*, 4269. [CrossRef]

45. Kushner, H.J.; Dupuis, P.G. *Numerical Methods for Stochastic Control Problems in Continuous Time*; Springer: New York, NY, USA, 1992. [CrossRef]

46. Fleming, W.H.; Soner, H.M. *Controlled Markov Processes and Viscosity Solutions*, 2nd ed.; Springer: New York, NY, USA, 2006. [CrossRef]

47. Puterman, M.L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*; Wiley-Interscience: New York, NY, USA, 2014.

48. Charalambous, C.D.; Ahmed, N. Equivalence of decentralized stochastic dynamic decision systems via Girsanov’s measure transformation. In Proceedings of the 53rd IEEE Conference on Decision and Control, Los Angeles, CA, USA, 15–17 December 2014; pp. 439–444. [CrossRef]

49. Telsang, B.; Djouadi, S.; Charalambous, C. Numerical Evaluation of Exact Person-by-Person Optimal Nonlinear Control Strategies of the Witsenhausen Counterexample. In Proceedings of the 2021 American Control Conference (ACC), New Orleans, LA, USA, 25–28 May 2021; pp. 1250–1255. [CrossRef]

50. Ruthotto, L.; Osher, S.J.; Li, W.; Nurbekyan, L.; Fung, S.W. A machine learning framework for solving high-dimensional mean field game and mean field control problems. *Proc. Natl. Acad. Sci. USA* **2020**, *117*, 9183–9193. [CrossRef]

51. Lin, A.T.; Fung, S.W.; Li, W.; Nurbekyan, L.; Osher, S.J. Alternating the population and control neural networks to solve high-dimensional stochastic mean-field games. *Proc. Natl. Acad. Sci. USA* **2021**, *118*, e2024713118. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.