Deep Reinforcement Learning With Reward Design for Quantum Control

Haixu Yu and Xudong Zhao

***Abstract*—Deep reinforcement learning (DRL) has been recognized as a powerful tool in quantum physics, where DRL's reward design is nontrivial but crucial for quantum control tasks. To address the problem of over-reliance on human empirical knowledge to design DRL's rewards, we propose a DRL with a novel reward paradigm designed by the learning process information (DRL-LPI), where the learning process information (LPI) comprises the state information and the experiences. In DRL-LPI, the state information after being classified by a fidelity threshold, and the experiences are first stored simultaneously in the respective sequences, and this process is repeated until a similar-segment ends. Then, the stored state information is converted to the real value and used to design the reward value by applying a self-amplitude function. Next, the designed reward values are integrated with the stored experiences to compose transitions for DRL's training. Through comparisons to five representative reward schemes, the proposed DRL-LPI is evaluated on two typical quantum control tasks, i.e., the spin-(1/2) quantum state control and many-coupled qubits state control, and the experimental results show the superior learning efficiency and control performance of the proposed approach. More results show that DRL-LPI exhibits the ability to learn the control strategy with few control actions compared to stochastic gradient descent (SGD) and genetic algorithm (GA).**

***Impact Statement*—Over the past few years, quantum machine learning has received growing attention. In particular, reinforcement learning (RL) and quantum physics have gradually intersected, and one representative aspect is that some impressive results have been achieved regarding the application of RL algorithms in quantum system tasks. Despite some advances, the full potential of RL remains massively unexplored in quantum physics. A major limitation is how to reward the learning agent. Most previous works adopted hand-designed methods, which tend to be time-consuming and vulnerable to human interference from empirical knowledge. The RL algorithm proposed in this paper reduces the above limitation, where rewards are automatically generated with the learning process. This paper is helpful to the technical research on automated reinforcement learning and quantum machine learning.**

***Index Terms*—Deep reinforcement learning (DRL), learning process information (LPI), quantum control, reward design.**

Manuscript received 15 September 2022; revised 25 October 2022; accepted 23 November 2022. Date of publication 28 November 2022; date of current version 15 March 2024. This work was supported in part by the National Natural Science Foundation of China under Grant U21A20477 and Grant 61722302, in part by the Fundamental Research Funds for the Central Universities under Grant DUT22ZD402, in part by the Liaoning Revitalization Talents Program under Grant XLYC1907140, and in part by the National Major Science and Technology Project under Grant 2019-V-0010-0105. This paper was recommended for publication by Associate Editor Francesco Piccialli upon evaluation of the reviewers' comments. *(Corresponding author: Xudong Zhao.)*

The authors are with the Key Laboratory of Intelligent Control and Optimization for Industrial Equipment, Ministry of Education, Dalian University of Technology, Dalian 116024, China (e-mail: haixuyuhcg@gmail.com; xdzhaohit@gmail.com).

Digital Object Identifier 10.1109/TAI.2022.3225256

I. Introduction

QUANTUM control is essential for achieving quantum computation [1], [2] and quantum communication [3], and, efficient quantum state manipulation is recognized as the core of establishing powerful quantum information technology [4], [5]. Quite recently, a series of quantum control methods have been developed [6]. For example, genetic algorithm (GA) [4], [7], [8] has been utilized to control different quantum systems. However, optimizing the performance of GA usually requires a large amount of experimental data, which is particularly time-consuming when solving problems involving complex quantum systems. Some gradient-based algorithms have also been proposed for quantum control, such as stochastic gradient descent (SGD) [9], gradient ascent pulse engineering [10], [11], chopped random basis [12], [13], and others [14], [15], [16]. However, these gradient-based methods usually fall into local optimum because of their algorithmic mechanisms. Furthermore, the gradient information of a quantum system may not be readily available in many practical applications, which brings many limitations to the abovementioned gradient-based methods. Controlling quantum systems, owing to the intrinsic complexity of quantum mechanics, is still an enormous challenge. Thus, it is important to explore alternative tools that are not excessively dependent on perfect system models for quantum control, where machine learning may play an important role.

Reinforcement learning (RL) is a model-free method without needing the gradient information of system dynamics [17]. As one of the most promising active learning techniques in machine learning [18], RL learns by trial-and-error and often achieves an outstanding performance [19], [20], [21], [22], [23], [24]. Different from supervised learning that needs a large amount of labeled input data, RL requires relatively few manual inputs and performs well with unlabeled data, which facilitates dealing with the uncertainty (or stochasticity) of physical systems. Recently, RL algorithms represented by deep reinforcement learning (DRL) have attracted increasing attention in some quantum system tasks, such as quantum state preparation [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], finding the ground states [36], [37], quantum gate control [38], [39], quantum error correction [40], and quantum circuit optimization [41], [42], [43], [44]. Despite recent advances, the full potential of RL remains massively unexplored in quantum physics. In particular, the vast majority of studies have focused on testing the RL's applicability to quantum control problems and ignored the fact that the feedback information of quantum systems is often limited and expensive, while few works have considered how to reward

the learning agent to efficiently solve quantum control problems. It is highly desirable to design an efficient and workable RL algorithm for quantum control based on the characteristics of quantum systems.

When applying RL to solve a quantum control problem, several unique challenges naturally arise because of the complex state space and dynamic characteristics of the controlled quantum system. One extremely challenging task is designing RL's rewards for the agent. The RL algorithm obtains an optimal control strategy by maximizing the accumulated rewards, where the rewards imply the unknown quantum control task and influence the learning efficiency (even success) of RL. In practical settings, the state of a quantum system is usually partially observable to the agent and many quantum states generated during the learning process may not provide useful reward values, which hinders the RL's application performance in quantum control. It is, therefore, necessary to investigate how to design effective reward values for RL. However, reward design of RL in quantum control has not received enough attention in existing studies. A statistical review of relevant papers reveals that most existing reward schemes are artificially (or empirically) designed for specific situations (detailed discussions of the existing reward schemes will be provided in the subsequent section of Relevant Works), which will cause the RL's application effectiveness to be greatly influenced by artificial experience and may further affect the RL's potential in quantum control.

In fact, the rewards have been recognized as a critical source for an agent to learn intelligent behavior [17], [45], and RL's reward design has gradually attracted the interest of many researchers in the field of artificial intelligence. Some representative studies have been got, such as reward shaping [46], potential-based reward [47], intrinsic reward [48], emotion-based reward [49], and inverse RL [50]. Unfortunately, existing reward design methods cannot be directly applied to quantum control tasks, because most of them increase the requirement for environmental feedback information, which is not always allowed or cannot be realized for quantum systems. In fact, the feedback information of a quantum system is usually obtained in the form of measurements, which is a precious feedback signal, e.g., at the cost of consuming numerous quantum state copies in practical experiments. Considering the preciousness of the obtained information about quantum systems, we propose a new reward design approach with the goal of well using the quantum system feedback information.

In this article, we focus on the learning process information (LPI) of quantum systems and propose a DRL with a novel reward paradigm designed by the learning process information (DRL-LPI) with the purpose of autogenerating rewards to efficiently solve quantum control problems. Specifically, the proposed DRL-LPI method is achieved by the following three main aspects: LPI record, generation criterion, and reward generation. As for the steps of LPI record, the experience at each time step needs to be stored as a sequence. In addition, the state information at each time step is classified by a fidelity threshold and then also stored as a sequence, which will be used to design the reward value and integrated into the transition to train DRL. In order to improve the production rate of the integrated transitions, a similar-segment is defined as the condition of the reward generation process. As for the steps of reward generation, the data type of the stored state information is first transformed into the real value. And then, a self-amplitude function is defined to design the reward value based on the stored state information after numerical type conversions. Next, the designed reward values will be integrated with the stored experiences to compose the effective transitions for training DRL. In summary, the main contributions of this article are listed as follows.

1) A DRL with a novel reward paradigm designed by the learning process information (DRL-LPI) is proposed for quantum control. LPI is formally described, and a self-amplitude function is designed so that the stored LPI can be automatically converted into reward values without human interference.
2) The proposed DRL-LPI method is applied to the spin-(1/2) quantum system and many-coupled qubits. Comparative numerical results with SGD, GA, and five existing reward schemes are presented to demonstrate the superior performance of DRL-LPI.
3) Compared with the existing reward schemes, the proposed DRL-LPI method does not increase the requirement for quantum feedback information, and even has the advantage of reducing the requirements in terms of feedback times and accuracy.

The rest of this article is organized as follows. Section II introduces the basic concepts about RL, the quantum control problem, and relevant works. In Section III, we first present the framework of DRL-LPI, followed by the definitions of LPI record and reward generation, and then give the final integrated implementation. Experiments on the spin-(1/2) quantum system and many-coupled qubits are conducted in Section IV. Finally, Section V concludes this article.

Notation: S denotes the state space; A denotes the action space; P denotes the state transition probability; R is the reward function; T denotes the total time per epoch; π is the policy function; s_t, a_t r_t, and F_t denote the state, the action, the reward, and the fidelity at a time step t, respectively; γ is the discount factor; θ denotes the neural network parameter; $\hbar$ is the reduced Plank constant; $H(t)$ is the time-dependent Hamiltonian; $u_m(t)$ denotes the m th time-dependent control field; U is the unitary operator; δt is the time duration for each piece-wise unitary; SS denotes the similar-segment; e denotes the experience; E denotes the similar-segment experience pool; d denotes the state information; D denotes the similar-segment state information pool; d_{thre} is the fidelity threshold used to distinguish the type of state information; F_{thre} is the fidelity threshold used to determine the ending condition of a similar-segment; FI denotes the fidelity value interval of a similar-segment; TCI$(\cdot)$ is the trans-code information function; SAII$(\cdot)$ is the self-amplitude information-index function; ID$(\cdot)$ is the integration data function; r_v denotes the designed reward with the storage index v; $\mathbb{E}$ denotes the mathematical expectation; $\mathbb{R}$ is the set of all real number; $|\psi_0\rangle$ denotes the initial quantum state; $|\psi_{\text{target}}\rangle$ and $|\psi_{\text{final}}\rangle$ are the target quantum state and the final quantum state, respectively; $f(\cdot)$ denotes the quantum state transformation function for transforming $|\psi\rangle$ into s.

II. Preliminaries and Relevant Works

A. Reinforcement Learning

The basic formalization of RL is based on the model of Markov decision process (MDP). A tuple $\langle S, A, P, R\rangle$ is usually used to describe an MDP, where S is the space of all possible state, A is the action space, $P : S \times A \times S \to P(S)$ is the state transition probability, $R : S \times A \to \mathbb{R}$ is the reward function. The learning process of RL can be described as a state-action-reward sequence. At a time step t, the agent observes the state $s_t \in S$. Then, the agent takes an action $a_t \in A$ based on the policy $a_t = \pi(s_t)$ and gets a reward value r_t with s_t being transmitted into the next state s_{t+1}. Thus, the transition (s_t, a_t, r_t, s_{t+1}) at a time step t can be created. The learning goal of the RL's agent is to maximize the cumulative discounted future rewards $R_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$, where $\gamma \in [0, 1]$ is the discount factor. The expected sum of discounted rewards corresponding to a policy π can be described as

$$Q^{\pi}(s, a) = \mathbb{E}[R_t | s_t = s, a]. \tag{1}$$

When RL is used to solve the problem with high-dimensional state space, a deep neural network called deep Q-network (DQN) with the parameter θ is usually used as a universal function approximator to approximate $Q(s, a)$, i.e., $Q(s, a; \theta) \approx Q(s, a)$. DQN is a famous DRL algorithm that can be trained by minimizing a sequence of loss functions

$$\text{Loss}(\theta) = (y - Q(s, a; \theta))^2 \tag{2}$$

where $y = r + \gamma \max_{a'} Q(s,' a'; \theta^-)$ is the target, s' is the next state generated by taking an action a at a state s, θ^- denotes the parameter of the Q-network, which is fixed during the computation of y and is updated after some training steps. Differentiate the loss function with respect to θ, and the gradient is formulated as

$$\nabla_{\theta}\text{Loss} = \left[r + \gamma \max_{a'} Q(s,' a'; \theta^-) - Q(s, a; \theta)\right] \nabla_{\theta} Q(s, a; \theta). \tag{3}$$

B. Quantum Control Problem

Consider a general control problem of a closed quantum system, the time evolution of a quantum state $|\psi(t)\rangle$ can be governed by the Schrödinger equation

$$\iota\hbar \frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle \tag{4}$$

where $\iota = \sqrt{-1}$, $\hbar$ is the reduced Plank constant. $H(t)$ is the time-dependent Hamiltonian used to characterize the dynamics of the quantum system, and it can be expressed as $H(t) = H_0 + \sum_{m=1}^{M} u_m(t) H_m$, where H_0 is the time-independent free evolution part, $u_m(t)$ is the mth time-dependent control field outside the system, and H_m couples the control field. The solution of (4) can be given by a unitary operator $U(t)$

$$\begin{cases} \iota \frac{d}{dt} U(t) = H(t) U(t) \\ U(t = 0) = I, t \in [0, T] \end{cases} \tag{5}$$

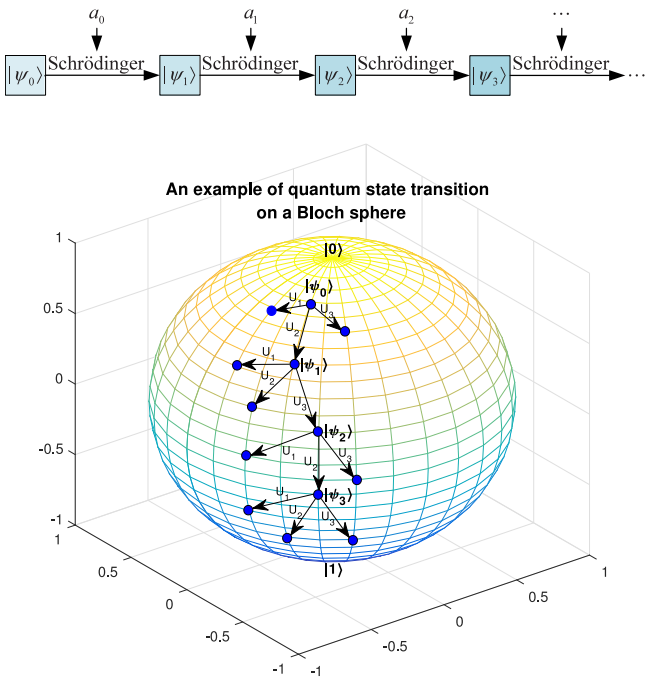

Fig. 1. MDP diagram for a quantum system.

where T is the total time. For an initial quantum state $|\psi(t_0)\rangle = |\psi(t = 0)\rangle$, an evolved quantum state can be got by $|\psi(t)\rangle = U(t)|\psi_0\rangle$. To prepare different quantum states, more sophisticated unitary is usually needed by applying N different piecewise unitaries: $U = U^N U^{N-1}, \ldots, U^1$, where each piece-wise unitary is $e^{-\iota(H_0 + \sum_{m=1}^{M} u_m(t) H_m)\delta t}$ with the time duration $\delta t = T/N$. A quantum control problem can be solved by optimizing $u_m(t)$, which is reflected by maximizing the overlap between the target state $|\psi_{\text{target}}\rangle$ and the evolved state $|\psi(t)\rangle$, i.e., maximizing the fidelity value $F(t) = |\langle\psi_{\text{target}}|\psi(t)\rangle|^2$.

In order to apply DRL to quantum control, the quantum control task to be solved in this article will be formulated as a model-free sequential decision-making problem (i.e., MDP). Here, to better illustrate that the learning process of a quantum system can be Markovian, Fig. 1 presents a simple example of quantum state transition on a Bloch sphere. In particular, the learning process can be divided into N discrete time steps. At each time step t, the agent receives a quantum system state $|\psi(t)\rangle$ and produces a control action a_t (i.e., $U(t)$), then, $|\psi(t)\rangle$ is evolved to the next quantum system state $|\psi(t + 1)\rangle$ following the Schrödinger equation described in (4) and the agent obtains a reward. It can be found that the next state $|\psi(t + 1)\rangle$ is related to the current state $|\psi(t)\rangle$ and action a_t, and independent of the past states. In the process of learning a quantum control strategy using DRL, the agent will interact with the quantum system multiple times to achieve the goal of maximizing the accumulated rewards, which reflects the importance of designing DRL's reward values for solving quantum control problems.

Note that, a quantum state $|\psi(t)\rangle$ usually needs to be fed into the DRL's neural network through some coding techniques, and the transformation form adopted in this article is $s_t = f(|\psi(t)\rangle) = [\text{Re}(|\psi_{ij}(t)\rangle), \text{Im}(|\psi_{ij}(t)\rangle)]$, where $|\psi_{ij}(t)\rangle$ is the

matrix element of $|\psi(t)\rangle$, and $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ are the functions taking the real and imaginary parts of a complex number. In the following, we let a_t denote a unitary operation $U(t)$ and let s_t denote the encoded state of $|\psi(t)\rangle$.

C. Relevant Works

To be informed of the existing reward design methods, some representative papers applying RL to quantum control are counted. In particular, we summarize the existing reward schemes into two basic groups, where the classification criterion is the relationship between reward values and fidelity, i.e., one is independent of fidelity and the other is dependent on fidelity.

The fidelity-independent reward schemes can be further summarized into two scenarios.

1) Multistage reward [25], [42]. In this case, the reward values are given in segments. More specifically, the reward values usually are artificially set to different nonzero real values corresponding to various fidelity value intervals.
2) Two-stage reward [26], [33], [51], [53]. The reward values are artificially set to two constant values in two situations, i.e., a relatively small value is set to the reward value if the target is not achieved, and a relatively large value if the target is achieved.

The fidelity-dependent reward schemes include the following three categories.

1) Staged reward with 0 [27], [38], [52]. In this scenario, the reward value is manually set to a fidelity-based function when the target is reached, otherwise it is set to 0.
2) Staged reward with nonzero value [32], [43]. Unlike the former scenario, the reward value is set to a nonzero value when the target is not reached.
3) Continuous reward [28], [29], [31], [34], [35], [41]. In this case, all reward values are given by an empirically designed fidelity-based function.

Despite the emergence of many reward design methods, there are several overlooked but serious problems. For example, existing reward schemes are almost empirical, which often leads to the unwelcome phenomenon of spending a lot of time determining how to divide the fidelity value interval and how to set the type (or size) of the reward value. In addition, some fidelity-dependent reward schemes are often infeasible in practice, because we cannot get an extremely precise fidelity value via existing measuring devices and methods. Unlike existing reward design schemes, we aim to make better use of the quantum system feedback information (i.e., LPI) to design reward values.

III. Reward Design of Deep Reinforcement Learning for Quantum Control

In this section, the framework of the proposed DRL-LPI method is first introduced. Then, some key functions used to achieve LPI record and reward generation are defined in detail, respectively. Finally, the integrated implementation of DRL-LPI for quantum control is given.

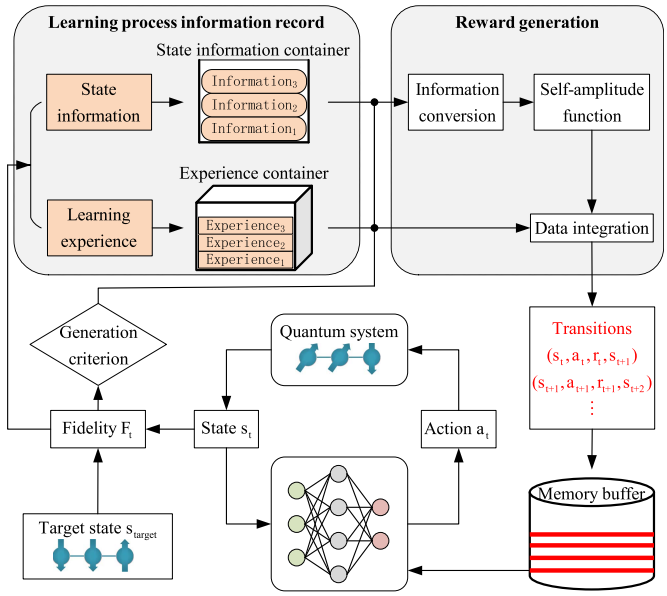

Fig. 2. Schematic of the proposed DRL-LPI method for quantum control.

A. Framework of DRL-LPI

When DRL is used to solve a quantum control problem, the agent characterized by the neural network will continuously interact with the quantum system to learn a control strategy. At a time step t, the agent observes the current quantum system state s_t and selects an action a_t, the quantum system will evolve to a new state s_{t+1}, then a fidelity value F_t for measuring the similarity between s_{t+1} and the target state s_{target} can be obtained. The transition, generated at each time step and stored in a memory buffer, is used to train the DRL's neural network to learn an effective quantum control strategy. As shown in Fig. 2, each transition comprises s_t, a_t, reward r_t and s_{t+1}, where the reward is important and indispensable for the DRL algorithm. Considering that the essential feature of DRL is to learn by maximizing accumulated reward values, the setting of reward values often determines the DRL's learning efficiency (even success). The general idea of the proposed DRL-LPI method is to design reward values according to the data generated during the interaction of the agent with the quantum system, and the data is called the LPI in this article, where two main steps of LPI record and reward generation are integrated to support our method.

The whole DRL-LPI is achieved by executing the following processes. First, two classes of LPI, the state information and the experience, are stored in a state information container and an experience container respectively in a sequence (relative definitions will be given in the following sections). Second, the stored state information is converted into the reward value by the information conversion process and a self-amplitude function, and then the designed reward values can be integrated with the stored experiences as transitions for training the DRL's neural network. It cannot be ignored that there is a generation criterion from the first step to the second step. Under our method, this

generation criterion is set to be related to the fidelity value between the quantum state at a time step and the target state. The abovementioned two major steps are continuously executed to generate enough transitions to train the DRL's neural network until an effective control strategy for the quantum control problem is obtained.

B. LPI Record

In order to obtain a control strategy for the quantum control problem with a target state, multiple interactions between the DRL's agent and the quantum system are often required. During the abovementioned learning process, once the number of performed actions reaches a certain value or the fidelity between a quantum state and the target state reaches the preset fidelity threshold, one-time learning process ends, which is called an epoch. Generally, the LPI generated in multiple epochs is often needed to learn a quantum control strategy. Therefore, the generated LPI is essential for solving the quantum control problem, and it serves as a reference for designing reward values in this study. Regarding that the fidelity has become an important metric for solving quantum control problems [1], [2], [6], we use the fidelity value as the condition to define a similar learning process for recording the generated LPI.

Definition 1. (Similar Learning Process): For an agent-environment interaction process characterized by a state-action-fidelity chain indexed by time series $s_0 \to a_0 \to F_0 \to, \ldots, \to s_{N-1} \to a_{N-1} \to F_{N-1} \to s_N$, a continuous set $\mathrm{SS} = \{s_t \to a_t \to F_t\}_{t=i}^{j}, (0 \le i \le j \le N-1)$ defines a similar learning process with the fidelity condition $F_t \in [F_{\min}, F_{\max}]$, where $F_{\min}$ and $F_{\max}$ are preset bounds.

We will take the end of each similar learning process (abbreviated as a similar-segment in the following contents) as the condition for generating reward values using the stored LPI. In particular, there are two kinds of LPI to be recorded during each similar-segment. One of them is the experience, which will be recorded in a similar-segment experience pool.

Definition 2. (Similar-Segment Experience Pool): Assume an experience consisting of state s, action a, reward r, and next state s' with a storage index v is expressed as $e_v = (s, a, r, s')_v$. The experience e_v $(v \in [v_1, v_2])$ generated in a similar-segment will be stored in a similar-segment experience pool E in a sequence, i.e., $E = \{e_{v_1}, \ldots, e_{v_2}\}$.

Remark 1: The reward r in each experience e is usually given an initial constant value (e.g., 0) that acts as a placeholder to store the designed reward value later.

Another LPI to be recorded is the state information of a quantum system. In this study, the state information at each time step is first classified as good or bad according to the numerical relationship between its corresponding fidelity value and a fidelity threshold, and then recorded in a similar-segment state information pool in the same way as the experience.

Definition 3. (Similar-Segment State Information Pool): Assume the state information corresponding to a quantum state $|\psi\rangle$ with a storage index v is expressed as d_v. The state information d_v $(v \in [v_1, v_2])$ generated in a similar-segment will be stored in a similar-segment state information pool D in a sequence, i.e., $D = \{d_{v_1}, \ldots, d_{v_2}\}$. In particular, d_v comprises the following two categories:

i) good state information: $d_v \leftarrow \mathrm{good}$, if $d_v \ge d_{\mathrm{thre}}$;

ii) bad state information: $d_v \leftarrow \mathrm{bad}$, if $d_v < d_{\mathrm{thre}}$;

where $d_v = |\langle\psi|\psi_{\mathrm{target}}\rangle|^2$ is the fidelity between $|\psi\rangle$ and the target state $|\psi_{\mathrm{target}}\rangle$, d_{thre} is the preset fidelity threshold.

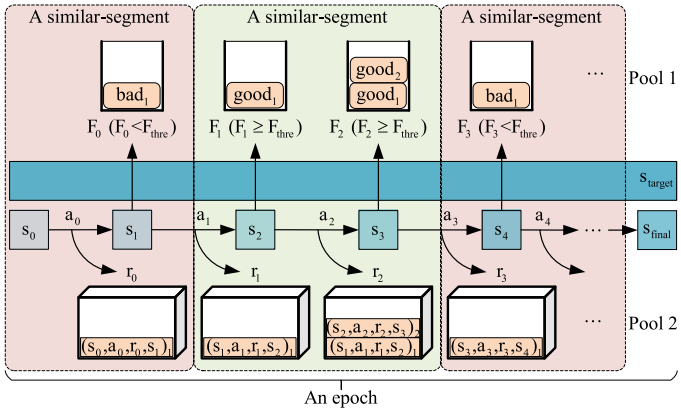

Fig. 3. Example of the LPI generation and storage in an epoch. $\mathrm{F_{thre}}$ is set as the fidelity condition of a similar-segment. $\mathrm{s_{final}}$ is the state at the end of an epoch, and $\mathrm{s_{target}}$ is the target state. Pool 1 and Pool 2 represent the similar-segment state information pool and similar-segment experience pool, respectively.

As shown in Fig. 3, an example of a learning epoch is given to intuitively show the LPI generation and storage process, in which a fidelity threshold d_{thre} is used to distinguish the type of state information and another fidelity threshold F_{thre} is used to determine the ending condition of a similar-segment (we set $F_{\mathrm{thre}} = d_{\mathrm{thre}}$ in Fig. 3 for simplicity). In our method, the fidelity threshold F_{thre} is considered as the only condition of a similar-segment. It can be seen from Fig. 3 that the condition of the first similar-segment is that the fidelity F_t $(t = 0, 1, 2, \ldots)$ corresponding to a evolved quantum state needs to be less than F_{thre}, while the second similar-segment is generated because F_1 no longer satisfies the fidelity condition of the first similar-segment. With the generation of a new similar-segment in an epoch, two pools (i.e., Pool 1 and Pool 2 in Fig. 3) start with the storage index 1 to record their respective LPI. During a similar-segment, both kinds of LPI are stored as sequences to ensure that the information stored in two pools corresponds to each other at the storage index v. Furthermore, one-step generating and storing LPI is summarized as Algorithm 1 consisting of two modules, one for setting the fidelity condition (i.e., fidelity value interval) of a similar-segment, and the other for generating and storing LPI corresponding to the current similar-segment. Therefore, as the agent interacts with the quantum system, the LPI generated at each time step will be recorded in the corresponding pool within a similar-segment recording period.

Remark 2: There may be multiple similar-segments in an epoch, and the stored LPI corresponding to the previous similar-segment should be cleared before the beginning of another similar-segment. In particular, two pools will be initialized when a similar-segment starts, and then used to store their corresponding LPI synchronously until this similar-segment ends.

Algorithm 1: One-Step Generating and Storing LPI in a Similar-Segment.

Input: time step t, state s_t, action a_t, reward r_t, next state s_{t+1}, fidelity F_t, similar-segment state information pool D with the fidelity threshold d_{thre}, similar-segment experience pool E, fidelity threshold of similar-segment F_{thre}, storage index v

Output: fidelity value interval of similar-segment FI, pools D and E

// Set fidelity value interval

if $F_n < F_{\text{thre}}$ **then**
  Set the fidelity value interval of the current similar-segment as FI $\leftarrow [0, F_{\text{thre}}]$
else
  Set FI $\leftarrow [F_{\text{thre}}, 1]$
end

// Generate and store LPI

Set the experience as $e_v = (s_t, a_t, r_t, s_{t+1})_v$
if $F_n < d_{\text{thre}}$ **then**
  Set the state information as $d_v \leftarrow \text{bad}_v$
else
  Set $d_v \leftarrow \text{good}_v$
end
Store d_v into D
Store e_v into E

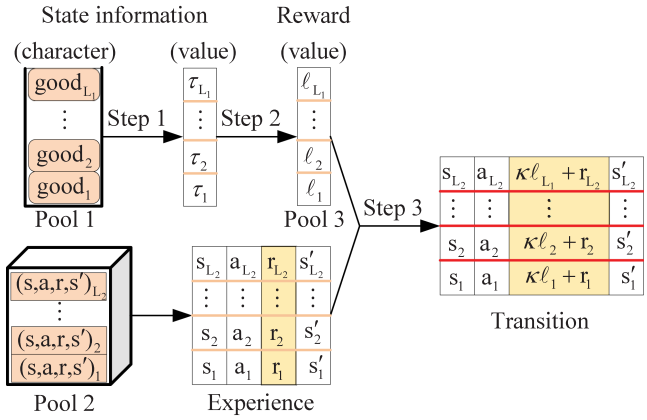

Fig. 4. Schematic of reward generation based on the stored LPI. As a display, a similar-segment with the state information type good is used. Pool 1, Pool 2, and Pool 3 represent the similar-segment state information pool, similar-segment experience pool, and similar-segment designed reward pool, respectively. Step 1 represents the trans-code information function TCI(good) $\rightarrow \tau$, Step 2 represents the self-amplitude information-index function SAII$(\tau_v, v) \rightarrow \ell_v$, and Step 3 represents the integration data function ID(L_1, κ, L_2, y_2) with $y_2 = 3$.

C. Reward Generation

Within the framework of the proposed DRL-LPI, a critical issue is how to use the stored LPI to design the reward value. Here, three functions are defined to address the abovementioned problem. At the end of each similar-segment, a similar-segment experience pool and a similar-segment state information pool can be constructed, and both of them have already stored their respective LPI in a sequence. In particular, there are at most two types of state information in the similar-segment state information pool, i.e., good and bad. In order to convert two kinds of the stored state information into the reward value, an information type conversion function (convert character types to numeric types) named the trans-code information function is first designed.

Definition 4. (Trans-Code Information Function): Assume a function TCI$(\nu) \rightarrow \tau$ ($\tau \in \mathbb{R}$) defines the transformation relation of the data type, it will take different real values for different kinds of character types.

With the trans-code information function, two types of state information stored in a similar-segment state information pool can be transformed into two real values. Note that each data in the similar-segment state information pool has a storage index value by default because the state information is stored as a sequence. After that, we define a self-amplitude information-index function to complete converting two real values into designed reward values.

Definition 5. (Self-Amplitude Information-Index Function): Suppose a list segment consisting of different real values, in which each data ξ has a storage index value v. SAII$(\xi_v, v) \rightarrow \ell_v$ ($\ell_v \in \mathbb{R}$) is defined as a function that the data value amplitude changes with the index value.

The designed reward values using the self-amplitude information-index function will be stored as a sequence, so a similar-segment designed reward pool with the stored index value can be got. Last but not least, an integration data function is defined to superimpose the similar-segment reward information pool into the similar-segment experience pool, so that we can get the transitions containing designed reward values corresponding to each similar-segment. The obtained transitions will be stored in a memory buffer and used to train the DRL's neural network to obtain an effective control strategy for the quantum control task, as shown in Fig. 2.

Definition 6. (Integration Data Function): Assume a list L_1 with X_1 rows and 1 column and a list L_2 with X_2 rows and Y_2 columns. ID(L_1, κ, L_2, y_2) is defined as the list data addition function that adds the data of L_1 to a specific column of L_2 at the ratio κ, which satisfies the conditions of $\kappa \in [0, 1]$, $X_1 = X_2$ and $y_2 = 1, \ldots, Y_2$.

To better understand the application order of three defined functions, an example of reward generation corresponding to a similar-segment is demonstrated in Fig. 4. In Fig. 4, Pool 1 and Pool 2 are the similar-segment state information pool and similar-segment experience pool, which have stored the state information and experience corresponding to a similar-segment, respectively. Note that, the storage index of Pool 1 is equal to that of Pool 2 (i.e., $L_1 = L_2$), which reflect the number of actions performed in a similar-segment. For a stored state information good$_v$ ($v = 1, 2, \ldots, L_1$), it will first become the designed reward r_v through the trans-code information function and the self-amplitude information-index function in turn, and then forms the transition $(s_v, a_v, \kappa\ell_v + r_v, s'_v)$ together with the stored experience $(s, a, r, s')_v$.

D. Integrated Implementation

When the proposed DRL-LPI method is used to solve a quantum control problem, the agent learns an effective control strategy by interacting with the quantum system through multiple epochs, and each epoch ends under the following two conditions. 1) When the number of actions executed by the agent reaches the maximum number, the state at this time step is called the final state $|\psi_{\text{final}}\rangle$. 2) When the fidelity between the state at a time step and the target state meets the preset fidelity condition, the state at this time step is called the target state $|\psi_{\text{target}}\rangle$. During the learning process of each epoch, the experience is stored in the similar-segment experience pool E, where the initial reward r_t that plays a placeholder role is set to 0. The state information is good or bad, which is stored in the similar-segment state information pool D. The abovementioned two storage processes are carried out simultaneously in the form of a sequence until a similar-segment SS ends. In particular, the ending condition of an SS follows two rules. 1) The agent reaches $|\psi_{\text{final}}\rangle$ or $|\psi_{\text{target}}\rangle$. 2) The fidelity at a time step t does not meet the condition of $F_t \in [F_{\min}, F_{\max}]$, where $[F_{\min}, F_{\max}]$ is the fidelity value interval corresponding to an SS.

Specifically, when a similar-segment SS$= \{s_t \to a_t \to F_t\}_{t=i}^{j}$ ends, an E which is a list of k rows and 4 columns is obtained and a D which is a list of k rows and 1 column is also formed, in which the state information d_v has the storage index value v and $k = j - i$. Based on the obtained E and D, some transitions with the designed reward values can be got by performing the following steps. First, for each d_v stored in D, the good is encoded as 1 and the bad is encoded as -1 with the trans-code information function

$$\tau_v = \text{TCI}(d_v) \tag{6}$$

where $\tau_v \in \{-1, 1\}$. Then, the designed reward value r_v can be got by using the self-amplitude information-index function

$$r_v = \text{SAII}(\tau_v, v). \tag{7}$$

According to whether an SS contains $|\psi_{\text{final}}\rangle$ or $|\psi_{\text{target}}\rangle$, the specific implementation of SAII(τ_v, v) is given in three cases. 1) For the case of $|\psi_{\text{final}}\rangle \notin$ SS and $|\psi_{\text{target}}\rangle \notin$ SS

$$\text{SAII}(\tau_v, v) = \tau_v \times (-|v - \text{floor}(k/2)| + \text{ceil}(k/2)) \tag{8}$$

where floor$(\cdot)$ represents a downward rounding function, i.e., floor$(w_1) = w_2$, where w_2 is the largest integer among all integers less than w_1. ceil$(\cdot)$ is a upward rounding function, i.e., floor$(w_1) = w_2$, where w_2 is the smallest integer among all integers greater than w_1. 2) For the case of $|\psi_{\text{final}}\rangle \in$ SS

$$\text{SAII}(\tau_v, v) = \tau_v \times (k/2). \tag{9}$$

3) For the case of $|\psi_{\text{target}}\rangle \in$ SS

$$\text{SAII}(\tau_v, v) = \begin{cases} \tau_v \times v & , v = 1, \ldots, k-1 \\ \tau_v + \tau_{\max} - \tau_{\min} & , \text{otherwise} \end{cases} \tag{10}$$

where $\tau_{\max}$ and $\tau_{\min}$ are the maximum reward value and the minimum reward value in an epoch, respectively. Then, the designed reward values r_v are stored in a sequence according to v to form a similar-segment designed reward pool R_v. Last, the transitions corresponding to an SS can be got by applying the integration data function

$$\text{transitions} = \text{ID}(R_v, \kappa, E, y_2) \tag{11}$$

where the ratio $\kappa \in [0, 1]$ and $y_2 = 3$.

Remark 3: In most situations, a similar-segment SS can contain only one of $|\psi_{\text{target}}\rangle$ or $|\psi_{\text{final}}\rangle$ because an SS will end once the agent reaches $|\psi_{\text{target}}\rangle$ or $|\psi_{\text{final}}\rangle$. A special situation may occur during the learning process, the agent happens to reach $|\psi_{\text{target}}\rangle$ when the number of executed actions is the maximum number, i.e., $|\psi_{\text{target}}\rangle$ happens to be $|\psi_{\text{final}}\rangle$ in an SS. For this special situation, we set the case of SAII(τ_v, v) as $|\psi_{\text{target}}\rangle \in$ SS by default.

With the abovementioned implementations, the transitions containing the designed reward values can be gradually generated during the learning process, which will be used to train the DRL's neural network until an effective quantum control strategy is got. The integrated DRL-LPI algorithm is summarized in Algorithm 3.

IV. Experiments

To test the performance of the proposed DRL-LPI algorithm for quantum control, several groups of experiments are carried out on the spin-(1/2) quantum system and many-coupled qubits.

A. Experimental Settings

We choose DQN as the benchmark DRL algorithm for solving quantum control problems. Specifically, the value function of DQN is approximated by a deep neural network with two 128-unit hidden layers, and the layers are connected by rectified linear unit activation function. Adam optimizer is used to train the neural network with the learning rate $\alpha = 1e-4$. The discount factor is $\gamma = 0.99$, and the memory size and the batch size are set as 100 and 32, respectively. During the learning process, $\epsilon-$greedy strategy is used to learn a control strategy, where the ϵ-greedy factor is initialized as $\epsilon = 0.9$ and determined by a hyperparameter λ. When the number of performed actions reaches the maximum action number N or the fidelity value reaches the preset fidelity value done, an epoch is ended, and a new epoch is started. As the first proof-of-principle demonstration of the porposed DRL-LPI method, we just take two classes of state information by a fidelity threshold d_{thre} (i.e., good and bad), and just consider two categories of similar-segments classified by a fidelity value F_{thre}. For all experiments, we set $d_{\text{thre}} = F_{\text{thre}} = 0.5$. The parameter κ of the self-amplitude information-index function ID(L_1, κ, L_2, y_2) is set as $\kappa = 1$ without trying other complex values that may be better.

Five representative reward schemes (marked as DRL-FIM, DRL-FIT, DRL-FDS0, DRL-FDS, and DRL-FDC) are compared to verify the effectiveness of DRL-LPI, which are summarized in Table I. To clearly show the learning performance of DRL-LPI, the above five reward methods are divided into two categories for comparison with DRL-LPI, respectively. (i) The fidelity-independent reward schemes (DRL-FIM and DRL-FIT) versus DRL-LPI. (ii) The fidelity-dependent reward schemes

Algorithm 2: Algorithm Description for DRL-LPI in Quantum Control.

Input: initial quantum state $|\psi_0\rangle$, target quantum state $|\psi_{\text{target}}\rangle$, maximum epoch number G, maximum action number per epoch N, batch size B, memory size M, initial ϵ-greedy value ϵ, decay factor λ, fidelity threshold of similar-segment F_{thre}, fidelity threshold of similar-segment state information pool d_{thrd}

Output: learned sequential action $a_i = \max_a Q(s_i, a)$

Initialize a DQN agent $Q(s, a|\theta)$ with a memory buffer $\mathcal{M}$, epoch number $g = 0$
while $g < G$ **do**
  Initialize a similar-segment state information pool D with a threshold d_{thrd}, a similar-segment experience pool E, fidelity value interval of similar-segment $\text{FI} = [0, 1]$, time step $t = 0$, storage index $v = 1$
  Decrease ϵ as $\epsilon = \epsilon \times \lambda$
  Transform $|\psi_t\rangle$ into s_t
  while $t < N$ **do**
    Choose action a_t at s_t using the exploration policy derived from $Q(s_t, a_t|\theta)$ (e.g., ϵ-greedy)
    Perform a_t, quantum state evolves to $|\psi_{t+1}\rangle$, obtain reward r_t and fidelity $F_t = F(|\psi_{t+1}\rangle, |\psi_{\text{target}}\rangle)$
    Transform $|\psi_{t+1}\rangle$ into s_{t+1}
    Obtain FI, D and E according to the LPI produced by v, t, s_t, a_t, r_t, s_{t+1}, F_n (Algorithm 1)
    if $|\psi_{t+1}\rangle$ *is not* $|\psi_{\text{target}}\rangle$ *or* $t + 1 \neq N$ **then**
      if $t = 0$ **then**
        Obtain FI, D and E according to the LPI produced by v, t, s_t, a_t, r_t, s_{t+1}, F_t (Algorithm 1)
      else
        if $F_t \in \text{FI}$ **then**
          $v \leftarrow v + 1$
          Obtain FI, D and E according to the LPI produced by v, t, s_t, a_t, r_t, s_{t+1}, F_t (Algorithm 1)
        else
          Transform the data type of d_v stored in D using $\tau_v = \text{TCI}(d_v)$
          Design reward using $r_v = \text{SAII}(\tau_v, v)$ as (8), obtain a similar-segment designed reward pool R_v
          Compose transitions by integrating R_v and E using $\text{ID}(R_v, \kappa, E, 3)$ as (11)
          Transfer transitions into memory buffer $\mathcal{M}$
          Clear all information stored in D and E, reset $v = 1$
          Obtain FI, D and E according to the LPI produced by v, t, s_t, a_t, r_t, s_{t+1}, F_t (Algorithm 1)
        end
      end
      $t \leftarrow t + 1$
    else
      $v \leftarrow v + 1$
      Obtain FI, D and E according to the LPI produced by v, t, s_t, a_t, r_t, s_{t+1}, F_t (Algorithm 1)
      Compose transitions using $\tau_v = \text{TCI}(d_v)$, $r_v = \text{SAII}(\tau_v, v)$ as (9)-(10) and $\text{ID}(R_v, \kappa, E, 3)$ as (11)
      Transfer transitions into memory buffer $\mathcal{M}$
    end
    if *Transition number in* $\mathcal{M}$ *is larger than batch size* B **then**
      Sample a batch of transitions and update the neural network parameter θ
    end
  end
  $g \leftarrow g + 1$
end

TABLE I
SUMMARY OF REWARD SCHEMES FOR COMPARATIVE EXPERIMENTS

Category	Reward scheme [ref]	**Reward value** (Fidelity condition)
fidelity-independent	DRL-FIM [25]	$r = \mathbf{0}(0 \leq Fid \leq 0.5) + \mathbf{10}(0.5 < Fid \leq 0.9) + \mathbf{100}(0.9 < Fid \leq done) + \mathbf{5000}(done < Fid \leq 1)$
	DRL-FIT [26]	$r = \mathbf{0}(0 \leq Fid < done) + \mathbf{1}(done \leq Fid \leq 1)$
fidelity-dependent	DRL-FDS0 [38]	$r = \mathbf{0}(0 \leq Fid < done) + (-\mathbf{log_{10}}(\mathbf{1} - \mathbf{Fid})(done \leq Fid \leq 1)$
	DRL-FDS [43]	$r =$ **(-0.01)**$(0 \leq Fid < done) +$ **(Fid-0.01)**$(done \leq Fid \leq 1)$
	DRL-FDC [29]	$r = \mathbf{Fid}^{\mathbf{8}}$

[1] Fid represents the fidelity value, $done$ represents the minimum fidelity value required to end an epoch. Notably, the ending condition of DRL-FDC is not clearly stated in the original paper, but we set it to the same as DRL-FIM, DRL-FIT, DRL-FDS0 and DRL-FDS in this paper

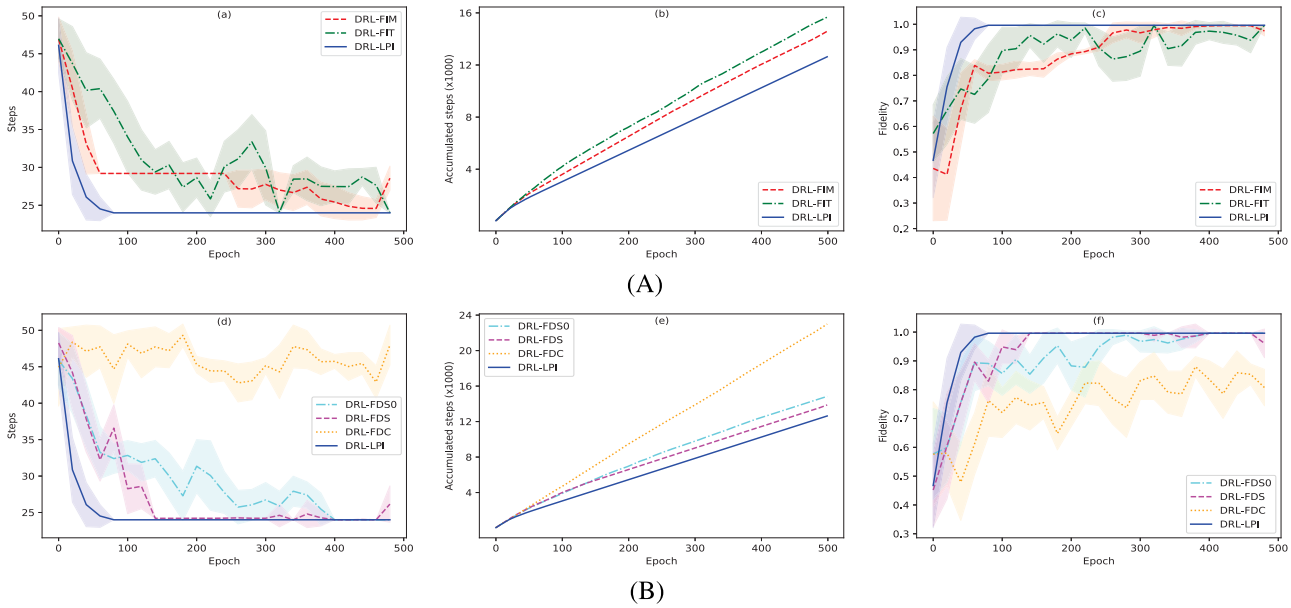

Fig. 5. Performance of DRL-LPI with comparison to five existing reward schemes (DRL-FIM, DRL-FIT, DRL-FDS0, DRL-FDS, DRL-FDC) on the spin-(1/2) quantum system. (A) Fidelity-independent reward schemes versus Our method. (a) Step convergence effect. (b) Accumulated steps. (c) Fidelity at the end of an epoch. (B) Fidelity-dependent reward schemes vrsus Our method, where (d), (e), (f) correspond to their counterparts of (a), (b), (c) in (A), respectively.

TABLE II
LIST OF HYPERPARAMETERS

Hyper-parameter	Spin-(1/2) system	Many-coupled qubits		
		$L=2$	$L=3$	$L=4$
G	500	100	1000	2000
N	50	20	20	20
λ	0.9	0.9	0.95	0.95
done	0.999 (DRL-FIM) / 0.99 (other cases)			0.9 (all cases)

[1]L represents the total number of spins.

(DRL-FDS0, DRL-FDS, and DRL-FDC) versus DRL-LPI. Note that, all experimental results are taken 5 runs to eliminate the randomness, in which each run is limited to the same maximum epoch number G to ensure the fairness of comparisons. The crucial parameters of different experiments are summarized in Table II. For each comparative experiment, three comparative indicators are the steps, the accumulated steps and the fidelity value, they are used to reflect the learning efficiency, the information feedback number of the quantum system, and the control performance with the learned control strategy, respectively. Note that, the curves of steps and fidelity are displayed in the form of mean value (solid lines) and standard error (shadow), where the upper and lower borders of shadow are obtained by the mean value plus standard error and the mean value minus standard error, respectively.

B. Experimental Results for the Spin-(1/2) Quantum System

We start with a simple yet physically relevant quantum control problem to illustrate the effectiveness of the proposed DRL-LPI method. Consider a spin-(1/2) quantum system characterized by the time-dependent Hamiltonian

$$H[J(t)] = 4\ J(t)\sigma_z + h\sigma_x \tag{12}$$

where σ_z and σ_x are the spin-(1/2) operators. The global control field can select three discrete values, i.e., $J(t) \in \{-1, 0, 1\}$. Here, we assume h is a constant and set $h = 1$ in this article. The general task refers to moving an initial quantum state $|\psi_0\rangle$ to the final quantum state $|\psi_{\text{final}}\rangle$ within the total control time T to maximize the fidelity between $|\psi_{\text{final}}\rangle$ and the target quantum state $|\psi_{\text{target}}\rangle$, i.e., $F = |\langle\psi_{\text{final}}|\psi_{\text{target}}\rangle|^2$. Specifically, we take $|\psi_0\rangle = |0\rangle$ and $|\psi_{\text{target}}\rangle = |1\rangle$, respectively. The total control time $T = \pi$ is divided into $N = 50$ equal time steps, and each time duration is $\delta t = T/N = \pi/50$. The quantum control problem of the spin-(1/2) quantum system is solved by obtaining an optimal control sequence.

The experimental results of DRL-FIM, DRL-FIT, and DRL-LPI on the spin-(1/2) quantum system are revealed in Fig. 5(A), it can be seen that our method has the fastest convergence speed, the minimum accumulated steps and the maximum fidelity compared to those of the other two reward schemes. Specifically, Fig. 5(a) reveals the learning efficiency reflected by steps (i.e., number of actions required for the control strategy learned at the end of each epoch), and it can be found that our method takes the minimal steps compared with DRL-FIM and DRL-FIT for each epoch, which leads to DRL-LPI spending less accumulated steps among 500 epochs shown in Fig. 5(b). It can be seen from Fig. 5(c) that our method tends to obtain a better fidelity within a shorter learning period, which reflects the high efficiency of DRL-LPI for this quantum control problem. Moreover, the resulting curve corresponding to our method has less shadow, while the curves of the other two reward schemes are shaded with a longer learning period in Fig. 5(a) and (c). Note that,

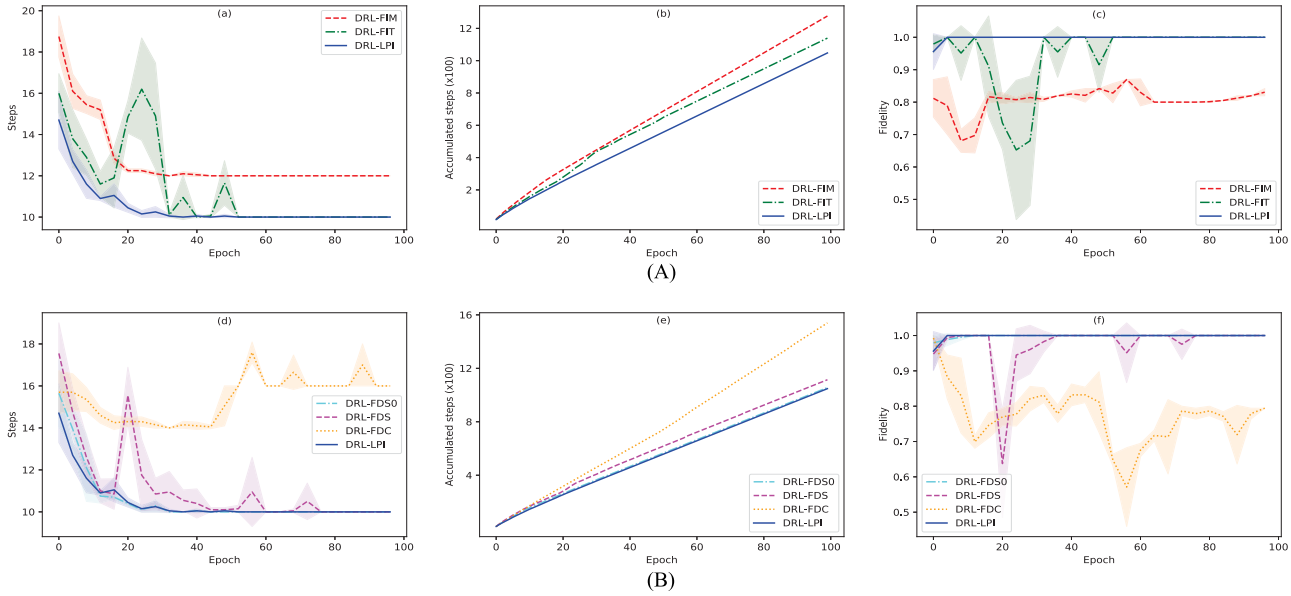

Fig. 6. Performance of DRL-LPI with comparison to five existing reward schemes (DRL-FIM, DRL-FIT, DRL-FDS0, DRL-FDS, DRL-FDC) on many-coupled qubits ($L = 2$). (A) Fidelity-independent reward schemes versus Our method. (a) Step convergence effect. (b) Accumulated steps. (c) Fidelity at the end of an epoch. (B) Fidelity-dependent reward schemes versus Our method, where (d), (e), (f) correspond to their counterparts of (a), (b), (c) in (A), respectively.

the shadow of the resulting curve represents the fluctuations in adjacent epochs.

The comparisons of DRL-FDS0, DRL-FDS, DRL-FDC, and our method on the spin-(1/2) quantum system are demonstrated in Fig. 5(B). Compared to the other three reward schemes, the better learning efficiency of our method can be reflected by the smallest steps in Fig. 5(d), the smallest accumulated steps in Fig. 5(e) and a high fidelity achieved within the smallest learning period in Fig. 5(f). In addition, the better convergence performance of our method compared to the other three schemes can be reflected in Fig. 5(d) and (f), because the resulting plot of our method is less shaded than other reward schemes. The abovementioned comparative experimental results in terms of steps, accumulated steps, and fidelity prove the high learning efficiency and good convergence performance of the proposed DRL-LPI method.

C. Experimental Results for Many-Coupled Qubits

Here, a closed chain of L-coupled qubits is considered to test the effectiveness of DRL-LPI, whose time-dependent Hamiltonian can be described as follows:

$$H\left[J_l(t)\right] = C\sum_{l=1}^{L-1}\left(o_x^l o_x^{l+1} + o_y^l o_y^{l+1}\right) + \sum_{l=1}^{L} 2J_l(t)o_z^l \quad (13)$$

where L is the total number of qubits, o_x^l, o_y^l, and o_z^l are the lth qubit operators, C is the coupling strength between two nearest-neighbor qubits, which is assumed to be a constant and set to $C = 1$ in this article. $J_l(t)$ is the piecewise-constant control within the total control time T, which is applied at the lth qubit. The control field is restricted as $J_l(t)/C \in \{0, 40\}$ for each qubit. For the quantum control problem, the initial quantum state is set as the leftmost qubit being up and the others down ($\uparrow \cdots \downarrow$), while the target quantum state is set as the rightmost qubit being up and the others down ($\downarrow \cdots \uparrow$). The total control time is set as $T = (L-1)\pi/2$, which is divided into $N = 20$ equal time steps. The quantum control problem of L-coupled qubits is solved by obtaining an optimal control sequence for each qubit such that the initial quantum state can be controlled to the target quantum state as close as possible, and the quality of quantum state control is evaluated by the fidelity similar to the case of the spin-(1/2) quantum system.

First, a relatively simple case of many-coupled qubits (set $L = 2$) is considered to test the effectiveness of the proposed DRL-LPI method, where the experimental results are shown in Fig. 6. The comparative experimental results demonstrated in Fig. 6(A) and (B) reveal that our method outperforms both the fidelity-independent reward schemes and the fidelity-dependent reward schemes in terms of learning efficiency and convergence performance. Specifically, we can see from Fig. 6(a) and (b) that DRL-LPI costs little learning period to learn a control strategy and costs the smallest accumulated steps within the total learning period among all reward schemes, which reveals the high learning efficiency of our method. It is seen from Fig. 6(c) that our method tends to achieve a high fidelity within the smallest epoch number compared with the other two reward schemes. In addition, the better convergence performance of our method can be reflected by Fig. 6(a) and (c), because the resulting curve of DRL-LPI is not shaded after a certain epoch number while the curves of the other two reward schemes are shaded within a larger epoch number. In addition, the high learning efficiency and good convergence performance of our method can also be reflected from the demonstrations in Fig. 6(d)–(f).

Then, a relatively complex case of many-coupled qubits (set $L = 3$) is considered to test the effectiveness of our method. Fig. 7(A) demonstrates the results of the comparison experiment between our method and the fidelity-independent reward schemes, and Fig. 7(B) displays the comparisons between our method and the fidelity-dependent reward schemes. From Fig. 7(a) and (d), it is clear that our method learns a

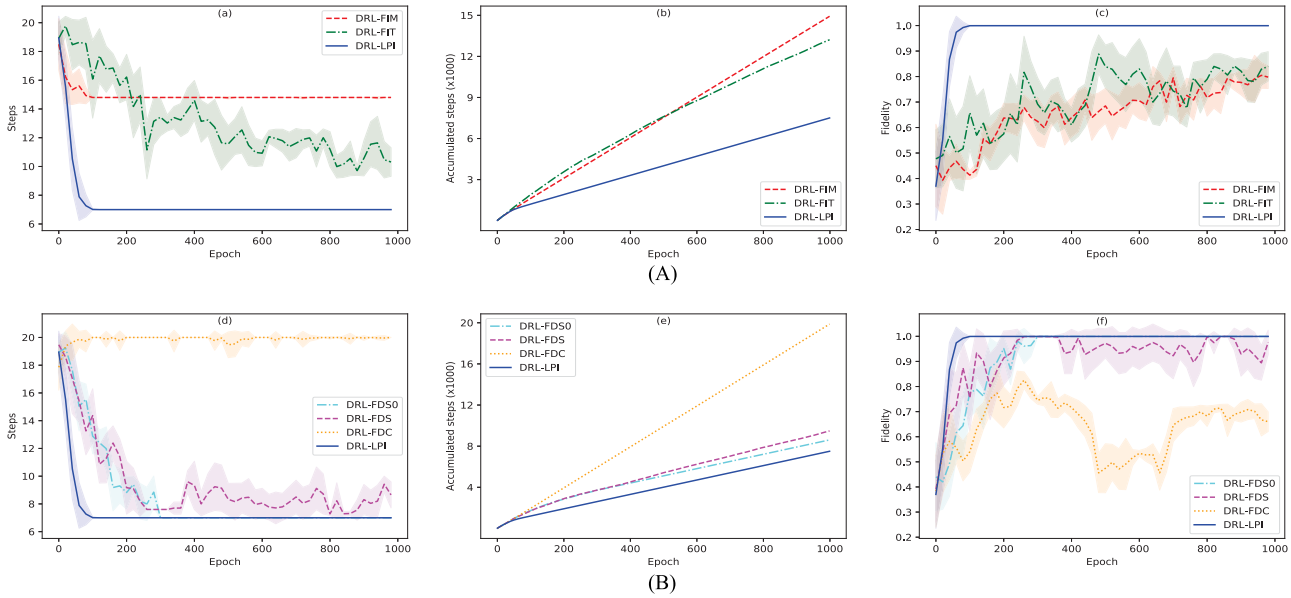

Fig. 7. Performance of DRL-LPI with comparison to five existing reward schemes (DRL-FIM, DRL-FIT, DRL-FDS0, DRL-FDS, DRL-FDC) on many-coupled qubits ($L = 3$). (A) Fidelity-independent reward schemes versus Our method. (a) Step convergence effect. (b) Accumulated steps. (c) Fidelity at the end of an epoch. (B) Fidelity-dependent reward schemes versus Our method, where (d), (e), (f) correspond to their counterparts of (a), (b), (c) in (A), respectively.

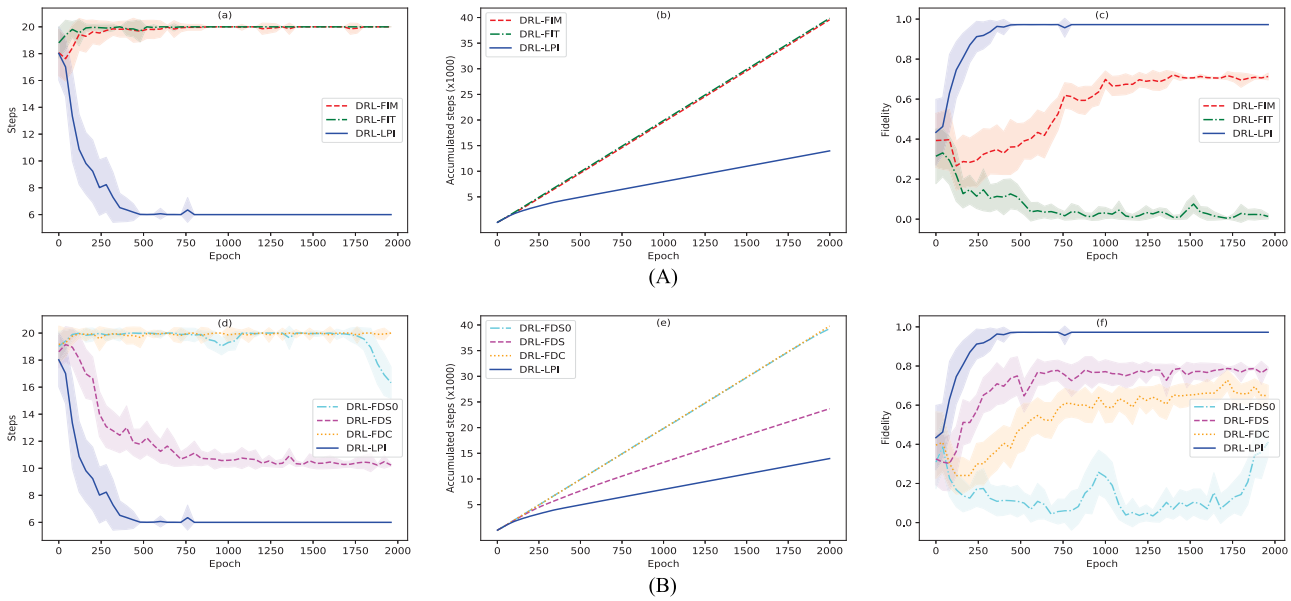

Fig. 8. Performance of DRL-LPI with comparison to five existing reward schemes (DRL-FIM, DRL-FIT, DRL-FDS0, DRL-FDS, DRL-FDC) on many-coupled qubits ($L = 4$). (A) Fidelity-independent reward schemes versus Our method. (a) Step convergence effect. (b) Accumulated steps. (c) Fidelity at the end of an epoch. (B) Fidelity-dependent reward schemes versus Our method, where (d), (e), (f) correspond to their counterparts of (a), (b), (c) in (A), respectively.

quantum control strategy with the shortest learning period. In addition, the proposed DRL-LPI method tasks the least accumulated steps among the total learning period compared to other reward schemes, which can be found in Fig. 7(b) and (e). Moreover, from Fig. 7(c) and (f), it is seen that our method tends to obtain a larger fidelity value than the other five reward schemes, which reflects our approach can better solve quantum control tasks.

Finally, a more complex case of many-coupled qubits (set $L = 4$) is considered to further verify the effectiveness of the proposed method. DRL-LPI is compared with the fidelity-independent reward schemes and the fidelity-dependent reward schemes, and the corresponding experimental results are shown in Fig. 8(A) and (B), respectively. It is clear that our method achieves the best performance among all comparative reward schemes regarding the steps in Fig. 8(a) and (d), the accumulated steps in Fig. 8(b) and (e), the fidelity value in Fig. 8(c) and (f). The abovementioned comparative experimental results reveal the high learning efficiency and good convergence performance of our method.

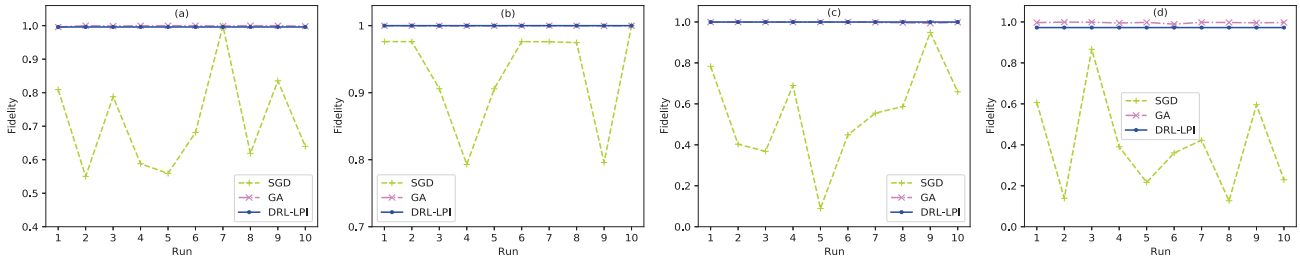

Fig. 9. Control performance comparisons of SGD, GA, and DRL-LPI in multiple experiments for four quantum system tasks. (a) Spin-(1/2) quantum system. (b)–(d) Many-coupled qubits take L values of 2–4 respectively.

D. Additional Comparative Experiments

The proposed DRL-LPI method aims to automatically generate rewards so that the quantum control problem can be solved efficiently. Here, some representative methods that have been used for quantum control, such as SGD [32] and GA [54] (see the Appendix for two algorithms), are compared to evaluate the effectiveness of DRL-LPI. The quantum control problems to be tested are to control the states of the spin-(1/2) quantum system and many-coupled qubits (L takes 2, 3, 4, respectively), and the related settings are the same as those in Sections IV-B and IV-C.

The comparison results of the abovementioned three methods (i.e., SGD, GA, and DRL-LPI) in multiple experiments (10 runs) are summarized in Fig. 9, where each marked point (i.e., fidelity value) is used to reflect the performance of a method for solving a quantum system task in one experiment. Fig. 9 shows that GA and DRL-LPI can solve quantum control tasks in all experiments and their curves are stable, while SGD cannot solve quantum control tasks in many experiments and its curve is turbulent. It is worth noting that GA in Fig. 9(d) shows a better control performance than DRL-LPI, because DRL-LPI has an end-of-epoch condition done, which limits its control performance, while GA does not have this additional restriction.

To show the learning details of Fig. 9, one of the experiments (Run 1) in Fig. 9(a)–(d) are displayed in Fig. 10(A)–(D), respectively. More specifically, for each quantum control task, the training curves (reflected by steps and accumulated steps) of the proposed DRL-LPI, GA, and SGD are demonstrated in Fig. 10(a) and (b), and the testing results (reflected by fidelity) based on the obtained control strategy are shown in Fig. 10(c). It can be seen from Fig. 10 that DRL-LPI spends the least accumulated steps in the training phase among three methods, and DRL-LPI can learn the control strategy that requires fewer control actions to solve quantum control problems.

E. Discussions

The purpose of this study is to explore the feasibility of using LPI to design reward values, so as to solve the problem of heavily relying on artificial experience to design DRL's reward values for quantum control tasks. Through the comparative experimental results shown in Figs. 5–8, it can be found that the proposed DRL-LPI method is capable of performing quantum

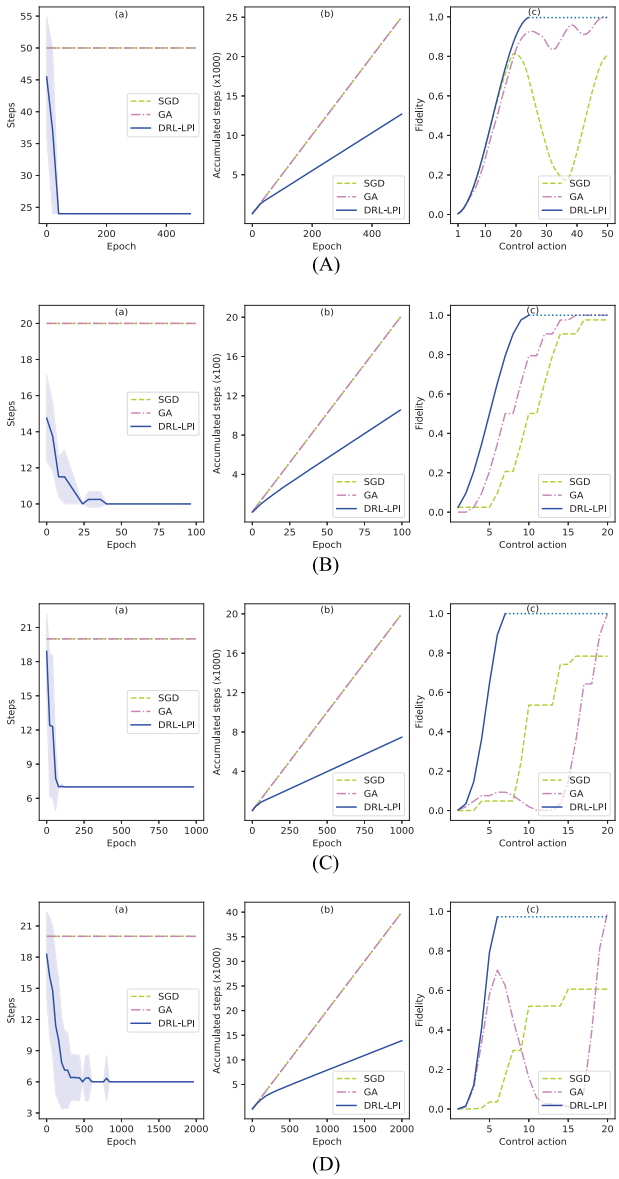

Fig. 10. Comparisons of SGD, GA, and DRL-LPI in one experiment for four quantum system tasks (A)–(D), which corresponds to Run 1 in Figs. 9(a)–(d), respectively. For each quantum system control task. (a) Step convergence effect. (b) Accumulated steps required by a method in the training phase. (c) Trajectories under the control strategy obtained by a method. (A) The spin-(1/2) quantum system. (B) Many-coupled qubits ($L = 2$). (C) Many-coupled qubits ($L = 3$). (D) Many-coupled qubits ($L = 4$).

state control tasks and outperforms even five representative reward schemes in terms of learning efficiency and control performance. One possible reason for the outstanding performance of our approach is that the stored LPI contains important knowledge and the reward paradigm generated by the stored LPI is beneficial for the DRL's agent to solve quantum control tasks.

Different from the existing reward schemes introduced in the section of Relevant Works, the reward values are designed based on the stored LPI in our method. Two significant advantages of the proposed method can be described as follows. On the one hand, DRL's reward values can be automatically generated with our method, which avoids relying on human empirical knowledge to carefully design rewards, such as designing the type (or size) of the reward value and dividing the fidelity value interval. On the other hand, our method is easier to implement in practice than many existing reward schemes in terms of the accuracy of feedback information from quantum systems. Considering the extremely accurate fidelity value of a quantum system is difficult to get in practice, our method uses a fidelity threshold 0.5 to classify the state information.

When applying a learning algorithm to solve a quantum control problem, there are two indicators that cannot be ignored. One is the efficiency of obtaining a quantum control strategy, and the other is the quality of the obtained strategy. In fact, some learning algorithms (e.g., SGD, GA, and DRL) usually learn a control strategy by continuously applying external control fields to a quantum system in the training phase, where the control field applied once to a quantum system can be simply called one step. Thus, the efficiency of using a learning algorithm to obtain a control strategy can be reflected by the number of accumulated steps. Generally, it is better to spend less accumulated steps to obtain an effective control strategy. One reason is that fewer accumulated steps mean fewer control resources (such as energy and time) are spent on obtaining an effective control strategy in practice. Another reason is that the control performed at each step will bring inevitable external interference to a quantum system, and thus, it is beneficial to maintain the stability of a quantum system by spending fewer accumulated steps. In addition, the quality of the learned control strategy can be evaluated by the fidelity value (determine whether a task is completed) and the required number of control actions. In general, a control strategy that can complete the control task is better than one that cannot. For two control strategies that can both complete the control task, the one with fewer control actions is better than the one with more control actions. It can be seen from Fig. 9 and 10 that DRL-LPI can efficiently obtain a better quantum control strategy than SGD and GA.

V. Conclusion

In this article, a DRL with a novel reward paradigm designed by the learning process information (DRL-LPI) is proposed for quantum control. In DRL-LPI, the reward values are automatically generated based on the stored LPI without requiring artificial experience. LPI consisting of the experience and the state information generated during the learning process is stored in a sequence, where the state information is first classified by a fidelity threshold and then stored. On this basis, the reward values are designed by a self-amplitude function acting on the stored state information. By integrating the designed reward values and the stored experiences, the effective transitions can be got for training the DRL's neural network. This integration process is determined by a similar-segment in our method. The experimental results demonstrate the superior performance of DRL-LPI by comparison with five representative reward shcemes. Moreover, the numerical results show the superior performance of DRL-LPI over SGD and GA in terms of obtaining an effective control strategy with fewer control actions. Our future work will try to define LPI in light of additional factors and consider the potential effects of LPI on DRL's reward design in quantum control. In addition, we will focus on investigations of the proposed reward design method for continuous quantum control tasks solved by other DRL algorithms.

Appendix

SGD is one of the most representative gradient-based optimization algorithms. The loss function of SGD can be defined as

$$\text{Loss}\left(\boldsymbol{J}\right) = 1 - \left|\langle\psi_{\text{target}}|U\left(\boldsymbol{J}\right)|\psi_0\rangle\right|^2 \tag{14}$$

where $|\psi_0\rangle$ and $|\psi_{\text{target}}\rangle$ represent the initial quantum state and the target quantum state, respectively. $U(\boldsymbol{J})$ is the evolution operator with the control sequence $\boldsymbol{J}$, which satisfies (5). The pseudocode of SGD is summarized in Algorithm 3. When using SGD to solve the quantum control problem, the learning rate β and upgrade rate η are set as $\beta = 0.01$ and $\eta = 0.5$, respectively.

GA is a random global optimization algorithm referring to the biological evolution mechanism in nature. When GA is used to solve a quantum system problem, the control sequence $\boldsymbol{J}$ can be coded as a binary string $\boldsymbol{BS}$ by the coding function $\text{CF}(\cdot)$. Each $\boldsymbol{BS}_i$ $(i = 1, 2, \ldots, \Gamma)$ can be called an individual in a population, where Γ is the population size. In particular, the individuals in the population are constantly evolving to produce better individuals to obtain an effective control strategy, and the fitness function is defined to evaluate the quality of the evolved individuals

$$\chi\left(\boldsymbol{J}\right) = \left|\langle\psi_{\text{target}}|U\left(\text{CF}\left(\boldsymbol{BS}\right)\right)|\psi_0\rangle\right|^2. \tag{15}$$

The specific evolution of GA is summarized in Algorithm 4. In this article, the population size Γ, the crossing-over rate P_c and the mutation rate P_m are set as $\Gamma = 50$, $P_c = 0.8$, and $P_m = 0.05$, respectively.

Note that, subject to the assumption of the tested quantum control problems that only discrete control values are allowed, the control sequence updated by SGD and GA needs to be discretized into the given discrete control values before it can be acted on the quantum system. Specifically, any control J_i

Algorithm 3: SGD Algorithm.

Input: maximum epoch number G, maximum action number per epoch N, upgrade rate η, learning rate β

Output: learned control sequence $\boldsymbol{J}$

Initialize a control sequence $\boldsymbol{J}=[J_1, J_2, ..., J_N]^T$ randomly, epoch number $g = 0$

while $g < G$ **do**

  Initialize a random unit vector $\boldsymbol{v}$

  Generate two control sequences $\boldsymbol{J}' = \boldsymbol{J} + \eta\boldsymbol{v}$, $\boldsymbol{J}'' = \boldsymbol{J} - \eta\boldsymbol{v}$

  Calculate the gradient with the loss function (14) $\boldsymbol{JJ} = \frac{\mathrm{Loss}(\boldsymbol{J}') - \mathrm{Loss}(\boldsymbol{J}'')}{2\eta}$

  Update the control sequence $\boldsymbol{J} \leftarrow \boldsymbol{J} - \beta\boldsymbol{JJ}$

end

Algorithm 4: GA Algorithm.

Input: Maximum iterations Z, population size Γ, control dimension per population N, crossing-over rate P_c, mutation rate P_m

Output: learned control sequence $\boldsymbol{J}$

Generate Γ binary strings $\boldsymbol{BS}$ randomly to form the initial population $\mathbb{S} = \{\boldsymbol{BS}_1, \boldsymbol{BS}_2, \ldots, \boldsymbol{BS}_\Gamma\}$, initialize iteration number $z = 0$

while $z < Z$ **do**

  Map $\boldsymbol{BS}_i$ to a control sequence $\boldsymbol{J}_i = \mathrm{CF}(\boldsymbol{BS}_i)$ $(i = 1, 2, \ldots, \Gamma)$

  Calculate fitness value $\chi_i = \chi(\boldsymbol{J}_i)$ with the fitness function (15) and sort χ_i $(i = 1, 2, \ldots, \Gamma)$

  Select the first $[\Gamma(1 - P_c)]$ binary strings in $\mathbb{S}$ with the highest fitness values, to form a new population $\mathbb{S}_1$

  Sample ΓP_c binary strings from $\mathbb{S}$ with probability $P(\boldsymbol{J}_i) = \chi_i \Big/ \sum_{j=1}^{\Gamma} \chi_j$, to form another population $\mathbb{S}_2$

  All binary strings in $\mathbb{S}_2$ are crossed in pairs, each of which is only executed once to update the binary strings in $\mathbb{S}_2$

  Perform mutation operation for the binary strings in $\mathbb{S}_2$ with probability P_m

  Update $\mathbb{S}$ by combining $\mathbb{S}_1$ and $\mathbb{S}_2$, $\mathbb{S} \leftarrow \mathbb{S}_1 + \mathbb{S}_2$

end

Map the binary string $\boldsymbol{BS}$ corresponding to the highest fitness as the learned control sequence $\boldsymbol{J} = \mathrm{CF}(\boldsymbol{BS})$

$(i = 1, 2, \ldots, N)$ in the control sequence $\boldsymbol{J}=[J_1, J_2, \ldots, J_N]^T$ updated by SGD or GA can be discretized as

$$J^j = \arg\min_{J^j} \left\{ |J_i - J^j| \right\} \tag{16}$$

where J^j comes from the given control set $\{J^j | j = 1, 2, \ldots\}$ of the considered quantum control problem.

REFERENCES

[1] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[2] C. H. Bennett and D. P. DiVincenzo, "Quantum information and computation," *Nature*, vol. 404, no. 6775, pp. 247–255, 2000.

[3] N. Gisin and R. Thew, "Quantum communication," *Nature Photon.*, vol. 1, no. 3, pp. 165–171, 2007.

[4] H. Rabitz, R. de Vivie-Riedle, M. Motzkus, and K. Kompa, "Whither the future of controlling quantum phenomena?," *Science*, vol. 288, no. 5467, pp. 824–828, 2000.

[5] H. M. Wiseman and G. J. Milburn, *Quantum Measurement and Control*. Cambridge, U.K.: Cambridge Univ. Press, 2010.

[6] D. Dong and I. R. Petersen, "Quantum control theory and applications: A survey," *IET Control Theory Appl.*, vol. 4, no. 12, pp. 2651–2671, 2010.

[7] M. Tsubouchi and T. Momose, "Rovibrational wave-packet manipulation using shaped midinfrared femtosecond pulses toward quantum computation: Optimization of pulse shape by a genetic algorithm," *Phys. Rev. A*, vol. 77, no. 5, 2008, Art. no. 052326.

[8] V. C. Gregoric, X. Kang, Z. C. Liu, Z. A. Rowley, T. J. Carroll, and M. W. Noel, "Quantum control via a genetic algorithm of the field ionization pathway of a rydberg electron," *Phys. Rev. A*, vol. 96, no. 2, 2017, Art. no. 023403.

[9] C. Ferrie, "Self-guided quantum tomography," *Phys. Rev. Lett.*, vol. 113, no. 19, 2014, Art. no. 190404.

[10] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbrüggen, and S. J. Glaser, "Optimal control of coupled spin dynamics: Design of NMR pulse sequences by gradient ascent algorithms," *J. Magn. Reson.*, vol. 172, no. 2, pp. 296–305, 2005.

[11] T. Schulte-Herbrüggen, A. Spörl, N. Khaneja, and S. Glaser, "Optimal control-based efficient synthesis of building blocks of quantum algorithms: A perspective from network complexity towards time complexity," *Phys. Rev. A*, vol. 72, no. 4, 2005, Art. no. 042331.

[12] P. Doria, T. Calarco, and S. Montangero, "Optimal control technique for many-body quantum dynamics," *Phys. Rev. Lett.*, vol. 106, no. 19, 2011, Art. no. 190501.

[13] T. Caneva, T. Calarco, and S. Montangero, "Chopped random-basis quantum optimization," *Phys. Rev. A*, vol. 84, no. 2, 2011, Art. no. 22326.

[14] R. Chakrabarti and H. Rabitz, "Quantum control landscapes," *Int. Rev. Phys. Chem.*, vol. 26, no. 4, pp. 671–735, 2007.

[15] J. Roslund and H. Rabitz, "Gradient algorithm applied to laboratory quantum control," *Phys. Rev. A*, vol. 79, no. 5, 2009, Art. no. 053417.

[16] R.-B. Wu, B. Chu, D. H. Owens, and H. Rabitz, "Data-driven gradient algorithm for high-precision quantum control," *Phys. Rev. A*, vol. 97, no. 4, 2018, Art. no. 42122.

[17] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[18] E. Alpaydin, *Introduction to Machine Learning*. Cambridge, MA, USA: MIT Press, 2004.

[19] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[20] F. Tang, B. Niu, G. Zong, and N. Xu, "Periodic event-triggered adaptive tracking control design for nonlinear discrete-time systems via reinforcement learning," *Neural Netw.*, vol. 154, pp. 43–55, 2022.

[21] M. Mahmud, M. S. Kaiser, A. Hussain, and S. Vassanelli, "Applications of deep learning and reinforcement learning to biological data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2063–2079, Jun. 2018.

[22] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 2064–2076, Jun. 2020.

[23] Z. Yuan, G. Li, Z. Wang, J. Sun, and R. Cheng, "Rl-CSL: A combinatorial optimization method using reinforcement learning and contrastive self-supervised learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, to be published, doi: 10.1109/TETCI.2021.3139802.

[24] F. Zou, G. G. Yen, L. Tang, and C. Wang, "A reinforcement learning approach for dynamic multi-objective optimization," *Inf. Sci.*, vol. 546, pp. 815–834, 2021.

[25] X.-M. Zhang, Z. Wei, R. Asad, X.-C. Yang, and X. Wang, "When does reinforcement learning stand out in quantum control? A comparative study on state preparation," *NPJ Quantum Inf.*, vol. 5, no. 1, pp. 1–7, 2019.

[26] Z. T. Wang, Y. Ashida, and M. Ueda, "Deep reinforcement learning control of quantum cartpoles," *Phys. Rev. Lett.*, vol. 125, no. 10, 2020, Art. no. 100401.

[27] H. Ma, D. Dong, S. X. Ding, and C. Chen, "Curriculum-based deep reinforcement learning for quantum control," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2022.3153502.

[28] Z. An, H.-J. Song, Q.-K. He, and D. Zhou, "Quantum optimal control of multilevel dissipative quantum systems with reinforcement learning," *Phys. Rev. A*, vol. 103, no. 1, 2021, Art. no. 012404.
[29] R. Porotti, A. Essig, B. Huard, and F. Marquardt, "Deep reinforcement learning for quantum state preparation with weak nonlinear measurements," *Quantum*, vol. 6, pp. 747–761, 2022.
[30] J. Brown et al., "Reinforcement learning-enhanced protocols for coherent population-transfer in three-level quantum systems," *New J. Phys.*, vol. 23, no. 9, 2021, Art. no. 93035.
[31] S. Borah, B. Sarma, M. Kewming, G. J. Milburn, and J. Twamley, "Measurement-based feedback quantum control with deep reinforcement learning for a double-well nonlinear potential," *Phys. Rev. Lett.*, vol. 127, no. 19, 2021, Art. no. 190403.
[32] X.-M. Zhang, Z.-W. Cui, X. Wang, and M.-H. Yung, "Automatic spin-chain learning to explore the quantum speed limit," *Phys. Rev. A*, vol. 97, no. 5, 2018, Art. no. 52333.
[33] J. Mackeprang, D. B. R. Dasari, and J. Wrachtrup, "A reinforcement learning approach for quantum state engineering," *Quantum Mach. Intell.*, vol. 2, no. 5, pp. 1–14, 2020.
[34] G. H. Xue and L. Qiu, "Preparation of three-atom ghz states based on deep reinforcement learning," *Quantum Inf. Process.*, vol. 20, no. 7, pp. 1–17, 2021.
[35] R.-H. He, R. Wang, S.-S. Nie, J. Wu, J.-H. Zhang, and Z.-M. Wang, "Deep reinforcement learning for universal quantum state preparation via dynamic pulse control," *EPJ Quantum Technol.*, vol. 8, no. 1, pp. 29–44, 2021.
[36] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks," *Science*, vol. 355, no. 6325, pp. 602–606, 2017.
[37] D.-L. Deng, X. Li, and S. D. Sarma, "Quantum entanglement in neural network states," *Phys. Rev. X*, vol. 7, no. 2, 2017, Art. no. 21021.
[38] Z. An and D. Zhou, "Deep reinforcement learning for quantum gate control," *Europhysics Lett.*, vol. 126, no. 6, 2019, Art. no. 60002.
[39] M. Y. Niu, S. Boixo, V. N. Smelyanskiy, and H. Neven, "Universal quantum control through deep reinforcement learning," *NPJ Quantum Inf.*, vol. 5, no. 1, pp. 1–8, 2019.
[40] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, "Reinforcement learning with neural networks for quantum feedback," *Phys. Rev. X*, vol. 8, no. 3, 2018, Art. no. 031084.
[41] T. Fösel, M. Y. Niu, F. Marquardt, and L. Li, "Quantum circuit optimization with deep reinforcement learning," 2021, *arXiv:2103.07585*.
[42] M. Ostaszewski, L. M. Trenkwalder, W. Masarczyk, E. Scerri, and V. Dunjko, "Reinforcement learning for optimization of variational quantum circuit architectures," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 18182–18194, 2021.
[43] E.-J. Kuo, Y.-L. L. Fang, and S. Y.-C. Chen, "Quantum architecture search via deep reinforcement learning," 2021, *arXiv:2104.07715*.
[44] L. Moro, M. G. Paris, M. Restelli, and E. Prati, "Quantum compiling by deep reinforcement learning," *Commun. Phys.*, vol. 4, no. 1, pp. 1–8, 2021.
[45] D. Silver, S. Singh, D. Precup, and R. S. Sutton, "Reward is enough," *Artif. Intell.*, vol. 299, 2021, Art. no. 103535.
[46] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proc. Int. Conf. Mach. Learn.*, 1999, pp. 278–287.
[47] S. M. Devlin and D. Kudenko, "Dynamic potential-based reward shaping," in *Proc. 11th Int. Conf. Auton. Agents Multiagent Syst.*, 2012, pp. 433–440.
[48] N. Chentanez, A. Barto, and S. Singh, "Intrinsically motivated reinforcement learning," *Adv. Neural Inf. Process. Syst.*, vol. 17, pp. 1281–1288, 2004.
[49] H. Yu and P. Yang, "An emotion-based approach to reinforcement learning reward design," in *Proc. IEEE 16th Int. Conf. Netw., Sens. Control*, 2019, pp. 346–351.
[50] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2000, pp. 663–670.
[51] H. Yu, X. Xu, H. Ma, Z. Zhu, and C. Chen, "Control design of two-level quantum systems with reinforcement learning," in *Proc. IEEE 33rd Youth Acad. Annu. Conf. Chin. Assoc. Autom.*, 2018, pp. 922–927.
[52] M. Bukov, A. G. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, "Reinforcement learning in different phases of quantum control," *Phys. Rev. X*, vol. 8, no. 3, 2018, Art. no. 31086.
[53] C. Chen, D. Dong, H.-X. Li, J. Chu, and T.-J. Tarn, "Fidelity-based probabilistic Q-learning for control of quantum systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 920–933, May 2014.
[54] D. Dong, X. Xing, H. Ma, C. Chen, Z. Liu, and H. Rabitz, "Learning-based quantum robust control: Algorithm, applications, and experiments," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3581–3593, Aug. 2020.

Haixu Yu received the B.S. degree in mechanical design manufacturing and automation from Northeast Forestry University, Harbin, China, in 2017 and the M.S. degree in control engineering from Nanjing University, Nanjing, China, in 2020. He is currently working toward the Ph.D. degree in control theory and control engineering with the School of Control Science and Engineering, Dalian University of Technology, Dalian, China.

His current research interests include reinforcement learning, quantum control, and quantum machine learning.

Xudong Zhao received the B.S. degree in automation from the Harbin Institute of Technology, Harbin, China, in 2005 and the Ph.D. degree in control science and engineering from Space Control and Inertial Technology Center, Harbin Institute of Technology, in 2010.

He was a Lecturer and an Associate Professor with the China University of Petroleum, Beijing, China. In March 2013, he was with Bohai University, Jinzhou, China, as a Professor. In 2014, He was a Postdoctoral Fellow with the Department of Mechanical Engineering, The University of Hong Kong, Hong Kong. Since December 2015, he has been with the Dalian University of Technology, Dalian, China, where he is currently a Professor. His recent research interests include hybrid systems, multiagent systems, intelligent control, and control of aero engine.

Dr. Zhao is currently an Associate Editor for IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS: SYSTEMS, *Nonlinear Analysis: Hybrid Systems*, *Neurocomputing*, *International Journal of General Systems*, *ACTA Automatica Sinica*, *Assembly Automation*, and *Journal of Aeronautics*, etc. In addition, he has been granted as the Outstanding Reviewer for *Automatica*, *IET Control Theory and Applications*, and *Journal of the Franklin Institute*, etc.