Improving robustness of quantum feedback control with reinforcement learning

Manuel Guatto

Dipartimento di Ingegneria dell'Informazione, Universitá degli Studi di Padova, via Gradenigo 6B, 35131 Padova, Italy
and Forschungszentrum Jülich, Institute of Quantum Control (PGI-8), D-52425 Jülich, Germany

Gian Antonio Susto

Dipartimento di Ingegneria dell'Informazione, Universitá degli Studi di Padova, via Gradenigo 6B, 35131 Padova, Italy
and Human Inspired Technology Research Center, Universitá degli Studi di Padova, via Luigi Luzzatti, 4, 35121 Padova PD, Italy

Francesco Ticozzi

Dipartimento di Ingegneria dell'Informazione, Universitá degli Studi di Padova, via Gradenigo 6B, 35131 Padova, Italy
and Department of Physics and Astronomy, Dartmouth College, 6127 Wilder Laboratory, Hanover, New Hampshire 03755, USA

(Received 31 January 2024; accepted 18 June 2024; published 3 July 2024)

Obtaining reliable state preparation protocols is a key step toward practical implementation of many quantum technologies, and one of the main tasks in quantum control. In this work, different reinforcement learning approaches are used to derive a feedback law for state preparation of a desired state in a target system. In particular, we focus on the robustness of the obtained strategies with respect to different types and amount of noise. Comparing the results indicates that the learned controls are more robust to unmodeled perturbations with respect to simple feedback strategy based on optimized population transfer, and that training on a simulated nominal model retains the same advantages displayed by controllers trained on real data. The possibility of effective off-line training of robust controllers promises significant advantages toward practical implementation.

DOI: 10.1103/PhysRevA.110.012605

I. INTRODUCTION

The development of reliable control tools for state preparation and engineering of desired dynamics in target quantum systems is a critical step toward quantum information processing on a scale that is suitable for real world applications [1–4]. Arguably, the main tool leveraged in classical control theory to design robust control laws that are able to adapt to unforeseen conditions and steer classical systems toward the target behavior is *feedback*: measuring some key quantity in real time and using the acquired information allows, among other things, for disturbance rejection and noise mitigation [5]. In the quantum domain, measurements and feedback become nontrivial, as they introduce probabilistic evolution and back action into the picture: models and methods for quantum feedback have been extensively studied [6–19], as well as successfully demonstrated experimentally [20]. Nonetheless, the robustness advantage of feedback methods has been demonstrated analytically to hold for quantum implementations only in particular cases [21–25], mostly with respect to uncertainty on the initial state or delays in the feedback loop, with some general results on model perturbations just starting to be derived [26].

In this work, we investigate the potential role of data-based learning methods in deriving feedback strategy and their robustness with respect to unknown model perturbations. Reinforcement learning (RL) has already been considered as a control design tool for quantum systems in Refs. [27–55], toward both control and error-correction tasks. With respect to the existing results, we focus on *comparing* different techniques in order to investigate the following:

(1) The role of the *a priori* knowledge on the model, by using different training scenarios for the controller;

(2) The robustness of RL methods, also compared to basic controllers, by introducing unmodeled noise in the evolution;

(3) The role of the measurement accuracy in addressing the control task; and

(4) The viability of of RL based on nominal models, rather than experimental data.

We consider the last point to be particularly interesting, and often overlooked in the literature: the time needed for obtaining an amount of data sufficient enough to grant convergence for the RL method could be a reason alone to discourage the use of these methods in practical scenarios, where typically preparations and measurements take significant time to be performed.

The analysis we propose is based on numeric simulations on a test-bed system, focusing on feedback state-preparation problems: given a quantum system in contact with a Markovian environment and subjected to repeated measurements, design a feedback control law that implements unitary control action dependent on the measurement outcomes and the system knowledge, aimed to minimize the distance between a desired target state ρ_{target} and the actual state $\rho(T)$ at the end of a finite time horizon $[0, T]$. The model we consider is inspired by the dynamics of multilevel atoms, in which the task is to maximize the electron population in a certain energy level [56], as well as recent works on feedback methods for

©2024 American Physical Society

optical cavity [20,28,57]. The results of our analysis, reported in detail in Sec. VII, indicate that model-based RL feedback laws are indeed a natural candidate when modeling errors and noise are expected, and that an accurate measurement process is crucial to obtain high fidelity.

The paper is organized as follows: Sec. II introduces the general model for our discrete-time feedback system, and Sec. III specializes it to the system that will be simulated. A short review of the key ideas behind the RL methods we employ is provided in Sec. IV, while Sec. V details the basic control strategy as well as three learning scenarios we consider; two learning scenarios employ a quantum description of the system and the same RL method, based on a proximal policy optimization (PPO) algorithm, while the third does not rely on any *a priori* knowledge on the model and builds the control law based only on classical data, i.e., the output of the measurement. The numerical experiments we conducted are reported in Sec. VI and the main findings summarized in Sec. VII.

II. DISCRETE-TIME FEEDBACK STATE PREPARATION WITH NOISE AND GENERALIZED MEASUREMENTS

In this section, we present the mathematical model for the feedback state-preparation problem described above. We define the various elements of the model, introduce the notation, and illustrate the interplay between quantum operations, measurements, and control parameters. In this paper only finite-dimensional systems are considered.

For open quantum systems undergoing Markovian evolutions, the transition of the system's state in Schroedinger's picture is associated to a linear, completely-positive trace-preserving (CPTP) map [58]. These maps admit a Kraus representation,

$$\mathcal{E}(\rho_t) = \sum_k E_k \rho_t E_k^\dagger, \tag{1}$$

where the operators E_k satisfy $\sum_k E_k^\dagger E_k = I$. In our setting, quantum operations will be used to describe both the noise and the control actions.

We assume our system to undergo a time-homogeneous Markovian noisy dynamics associated with a CPTP map $\mathcal{N}_\alpha$, with the parameter α to weigh the amount of noise injected in the system. Such map is represented by the ensemble of Kraus operators $\{N_k^\alpha\}$.

The noise action is followed by a generalized measurement with a finite set of outcomes $l = 1, \ldots, m$, associated with a set of measurement operators M_l^ϵ, where ϵ is a parameter associated with how informative the measurements are. In particular, $\epsilon = 0$ will correspond to projective measurements. The operators are such that $\sum_l M_l^{\epsilon\dagger} M_l^\epsilon = I$. The probability $p(l_t)$ for a specific outcome l at time t is computed as

$$p(l_t) = \mathrm{tr}(M_l^{\epsilon\dagger} M_l^\epsilon \rho_t'). \tag{2}$$

Once the measurement outcome l_t is obtained, the postmeasurement state $\rho_{t+1|l_t}$ is updated as follows:

$$\mathcal{M}_{l_t,\epsilon}(\rho_t) = \rho_{t+1|l_t} = \frac{M_{l_t}^\epsilon \rho_t M_{l_t}^{\epsilon\dagger}}{\mathrm{tr}\left(M_{l_t}^{\epsilon\dagger} M_{l_t}^\epsilon \rho_t\right)}. \tag{3}$$

The measurements are followed by a unitary control action, conditioned on the last measurement outcome. The unitary control is assumed to take the form $U_\beta = e^{-iH_c(\beta)}$, with H_c a control Hamiltonian and β the control parameter. In the following we will consider feedback laws of the form $\beta_t = \phi(\rho_0, \vec{l}_{t-1})$, where $\vec{l}_{t-1}$ represents the sequence of outcomes up to time $t-1$ and the functional ϕ can be determined either analytically or via reinforcement learning. The unitary control superoperator then becomes $\mathcal{U}_\beta$:

$$\mathcal{U}_\beta(\rho) = U_\beta \rho U_\beta^\dagger. \tag{4}$$

The evolution of the system is then obtained by iterating these steps. From a physical viewpoint, we consider the actual time scales of the measurement and control processes to be faster than the noise ones, so we can neglect the noise while measurements and control are acting. The dynamical evolution, conditional on a sequence of measurements outcomes $\vec{l}_t$, is obtained as the composition of $\mathcal{N}_\alpha$, $\mathcal{U}_{\phi(\rho_0,\vec{l}_{t-1})}$, and $\mathcal{M}_{l_t,\epsilon}$:

$$\begin{cases} \rho(t+1) = \mathcal{M}_{l_t,\epsilon} \circ \mathcal{U}_{\phi(\rho_0,\vec{l}_{t-1})} \circ \mathcal{N}_\alpha[\rho(t)] \\ \rho(0) = \rho_0, \end{cases} \tag{5}$$

with ρ_0 the initial condition for the dynamics. While the concatenated dynamics is not a CPTP map, as it is not linear due to the conditioning, its expectation over the measurements outcomes is such, and takes the form

$$\rho(t+1) = \sum_{k,l} M_l^\epsilon U_{\phi(\rho_0,l_{t-1})} N_k^\alpha \rho(t) N_k^{\alpha\dagger} U_{\phi(\rho_0,l_{t-1})}^\dagger M_l^{\epsilon\dagger}. \tag{6}$$

The design of stabilizing control laws ϕ is aimed to obtain convergence of the quantum state $\rho(t)$ toward the target state ρ_{target} [59] independently of the initial condition, namely, $\forall\rho(0)$, we want the dynamics above to ensure:

$$\lim_{t\to\infty} \rho(t) = \rho_{\text{target}}, \tag{7}$$

where $\rho_{\text{target}}(t)$ is the desired target state at time step t.

In what follows, we consider instead the related problem of *state preparation in finite time*, where the control task is, for some fixed time T, to

$$\underset{\phi}{\text{minimize}}\; d[\rho(T) - \rho_{\text{target}}], \tag{8}$$

where d is a suitable norm or pseudodistance. A stabilizing law on the infinite control horizon then represents an approximate solution for the finite-time state preparation problem, with its accuracy improving for larger T.

In the subsequent sections, we propose different ways to build feedback strategies under different assumptions regarding the available information on the system, and compare their robustness and performance toward state preparation in finite time.

III. A TEST-BED SYSTEM

For our numerical analysis, we consider a three-level quantum system inspired by that of Ref. [57]. Define the Hilbert space of interest as $\mathcal{H} = \mathbb{C}^3$, and the basis vector as

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad |2\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}. \tag{9}$$

A. Noise

Two key noise models, namely, the depolarizing noise and the random permutation channel, are considered in the simulations presented in the next section.

The *depolarizing channel* is a fundamental noise model that describes the effects of "isotropic" random errors and disturbances in quantum systems. It can be defined through its action on a quantum state ρ as follows:

$$\mathcal{N}_\alpha(\rho) = \alpha\frac{I}{3} + (1-\alpha)\rho. \tag{10}$$

$\mathcal{N}_\alpha(\rho)$ represents the quantum state after the application of the depolarizing noise and $\alpha \in [0, 1]$ is a parameter that quantifies the strength of the noise.

The *amplitude damping channel* for a qutrit system is a nonunital noise which reduces the energy of the system due to an interaction with the environment. The amplitude damping channel is described by its Kraus representation $\{|0\rangle, |1\rangle, |2\rangle\}$, defined as follows:

$$N_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{1-\gamma_1} & 0 \\ 0 & 0 & \sqrt{1-\gamma_2-\gamma_3} \end{pmatrix},$$

$$N_{01} = \begin{pmatrix} 0 & \sqrt{\gamma_1} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad N_{12} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \sqrt{\gamma_2} \\ 0 & 0 & 0 \end{pmatrix},$$

$$N_{03} = \begin{pmatrix} 0 & 0 & \sqrt{\gamma_3} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \tag{11}$$

with the following constraint:

$$\begin{cases} 0 \leqslant \gamma_i \leqslant 1 \quad \forall i \in \{1, 2, 3\} \\ \gamma_2 + \gamma_3 \leqslant 1. \end{cases}$$

In particular, in the simulated system we will set the γ_i parameters in function of a single parameter α:

$$\begin{cases} \gamma_1 = 0 \\ \gamma_2 + \gamma_3 = \alpha \\ \gamma_2 = \gamma_3. \end{cases}$$

With *random permutation channel* or random qutrit rotation channel, we denote quantum noise that captures randomized cycling between quantum states. To describe this channel for our qutrit system, we utilize a set of cyclic permutation matrices:

$$N_0^\alpha = \sqrt{1 - \frac{2\alpha}{3}} I,$$

$$N_1^\alpha = \sqrt{\frac{\alpha}{3}} \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix},$$

$$N_2^\alpha = \sqrt{\frac{\alpha}{3}} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, \tag{12}$$

with the corresponding CPTP map defined as follows:

$$\mathcal{N}_\alpha(\rho) = \sum_k N_k^\alpha \rho N_k^{\alpha^\dagger}. \tag{13}$$

Similar to the depolarizing channel noise, the parameter α plays a crucial role in characterizing the random qutrit flip channel. It defines the probability of the qutrit system undergoing a flip operation. A larger value of α indicates a higher likelihood of state flips, while a smaller value corresponds to a reduced probability of flipping.

B. Measurements

The measurement process we consider is associated with a set of operators, denoted as $M_0^\epsilon, M_1^\epsilon, M_2^\epsilon$, which satisfy the completeness constraint:

$$I = \sum_{i=0} M_i^{\epsilon\dagger} M_i^\epsilon. \tag{14}$$

In our simulated quantum system, we consider an "imprecise" version of projective measurement, which provides useful yet incomplete information about the basis state in which the system is in. The ϵ parameter limits the amount of information provided by the measurements.

To represent this operator, we define its Kraus representation as follows:

$$M_0^\epsilon = \begin{pmatrix} \sqrt{1-2\epsilon} & 0 & 0 \\ 0 & \sqrt{\epsilon} & 0 \\ 0 & 0 & \sqrt{\epsilon} \end{pmatrix},$$

$$M_1^\epsilon = \begin{pmatrix} \sqrt{\epsilon} & 0 & 0 \\ 0 & \sqrt{1-2\epsilon} & 0 \\ 0 & 0 & \sqrt{\epsilon} \end{pmatrix},$$

$$M_2^\epsilon = \begin{pmatrix} \sqrt{\epsilon} & 0 & 0 \\ 0 & \sqrt{\epsilon} & 0 \\ 0 & 0 & \sqrt{1-2\epsilon} \end{pmatrix}. \tag{15}$$

Notice that (1) each basis state is invariant for conditioning on *any* output of the measurement, and (2) the variable ϵ introduces an error in detecting the correct basis state, which is correctly represented by the outcome with probability $1 - 2\epsilon$.

C. Unitary control and state-preparation problem

Recall from the previous section that the control action is associated with a unitary of the form

$$U_\beta(t) = e^{-iH_c(\beta_t)}, \tag{16}$$

where we recall that β_t represents the time-dependent control parameter, acting on a time scale much faster than the noise one. Hence, its action can be considered decoupled from the rest of the dynamics and generating impulsive unitary control actions. For our model, we consider

$$H_c(\beta_t) = i[\beta_t(a - a^\dagger)], \tag{17}$$

where β_t is a real-valued function of time with codomain between -1 and 1, and the operator a is given by

$$a = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}. \tag{18}$$

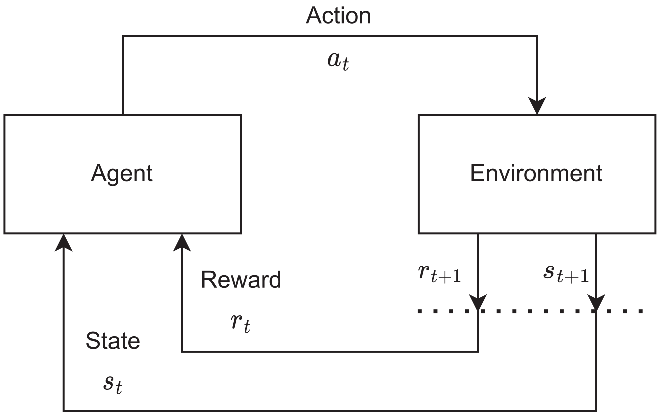

FIG. 1. Reinforcement learning interaction scheme.

To this unitary matrix corresponds the control superoperator $\mathcal{U}_\beta(\rho) = U_\beta \rho U_\beta^\dagger$ as defined before.

The target for designing the causal feedback law $\beta_t = \phi(\rho_0, \vec{l}_{t-1})$ is to obtain a state as close as possible to $\rho_{\text{target}} = |2\rangle\langle 2|$ at the end of the control period T. Notice that since the outcomes $\{l_t\}$ form a discrete-time stochastic process, both ρ and β_t become stochastic. More precisely, we aim to solve problem (8) with pseudo distance $d(\rho, \rho_{\text{target}}) = 1 - F(\rho, \rho_{\text{target}})$, where the fidelity is defined as

$$F(\rho, \rho_{\text{target}}) = \text{tr}\left[\sqrt{\sqrt{\rho}\rho_{\text{target}}\sqrt{\rho}}\right]^2. \tag{19}$$

IV. REINFORCEMENT LEARNING: A BRIEF INTRODUCTION

A. Essential elements

Reinforcement learning (RL) can be considered as a subfield of machine learning that focuses on training intelligent agents to make sequential decisions through interaction with their training environment (Fig. 1). Unlike supervised learning, where models are provided with labeled data, or unsupervised learning, which seeks to uncover hidden patterns in data, RL is focused on the learning of optimal *policies* by an *agent* in an *environment* that provides feedback in the form of rewards [60]. Optimality, in the context of RL, is associated with long-term objectives, aiming at maximizing a (weighted) sum of rewards in subsequent steps of a trajectory.

In the RL field, the state space, denoted as S, represents the set of all possible states in which an agent can be [61]. The state space can be discrete or continuous, and it captures all relevant information about the environment's current configuration. To determine the action that transitions the agent from s_t, the state at time t, to the next state s_{t+1}, the agent relies on a defined policy. Formally, this latter can be defined as a map that determines the probability distribution of selecting actions (a) in a given state (s). It is represented as $\pi(a|s)$. States in RL are considered Markovian, i.e., the current state contains all the relevant information for the agent and its dynamic, making the information about the past states irrelevant.

Moreover, we define a reward function, denoted as R, which quantifies the immediate consequence of the agent's action. Formally, $R(s, a)$ maps a state-action pair (s, a) to a real number, representing the reward obtained when applying action a from state s_{t+1}. Formally, the agent's goal is to maximize the expected cumulative reward, often expressed as the expected return, defined as $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$, where $\gamma \in [0, 1]$ represents a discount factor.

B. Policy gradient approach

In the realm of RL, Q-learning and policy gradient (PG) methods represent two distinct approaches. Q-learning estimates the value of state-action pairs and works well in discrete action spaces with fully observable environments. In contrast, PG directly optimizes policies, making it more effective for continuous action spaces and non-fully observable environments, where discretization and complete state information can be challenging for Q-learning. In this work we concentrate on PG methods: this section provides a brief introduction about the foundational principles of PG approaches.

We recall that a policy $\pi(a|s)$ determines the probability of selecting action a given the current state s of the RL environment. In the PG framework we assume that each policy is associated with a set of parameters θ.

As previously highlighted, the objective of RL lies in finding an optimal policy that maximize the expected return; in the PG framework this goal can be achieved by iteratively updating the θ parameters via the policy gradient RL update rule:

$$\delta\theta_j = \eta \frac{\partial \mathbb{E}[R]}{\partial \theta_j} = \eta \left[R \sum_t \frac{\partial}{\partial \theta_j} \ln \pi_\theta(a_t|s_t) \right]. \tag{20}$$

Here, η denotes the learning rate parameter, which is a real value parameter on which the converging proprieties of the RL algorithm depend (usually it takes values between 10^{-3} and 10^{-5}) and $\mathbb{E}[\cdot]$ represents the expectation value computed over all possible rewards. These fundamental elements form the core policy gradient approach. In practical applications, enhancements and refinements of the above equation are often employed.

Furthermore, Eq. (20) serves as the standard recipe for policy-based RL in fully observed environments. This approach can be extended to accommodate partially observed environments, where the policy relies solely on observations rather than the complete state information. These observations offer only partial insights into the actual state of the environment, introducing additional complexities in the RL framework.

In our work, we choose to adopt the Stable Baselines3 implementation [30] in which a neural network is used to compute the π_θ policy, with the multidimensional parameter (θ) encompassing all the network's weights and biases. The neural network takes the current state (s_t) as an input vector and produces the action probabilities (π_θ) as its output.

C. RL for feedback quantum control

This section aims to connect the general RL framework recalled above to the specifics of the problem at hand. An in-depth discussion of the details will be provided in Sec. V. In a quantum feedback control problem, quantum states and their dynamics are manipulated using control operations from some available set, while adapting the strategies depending on

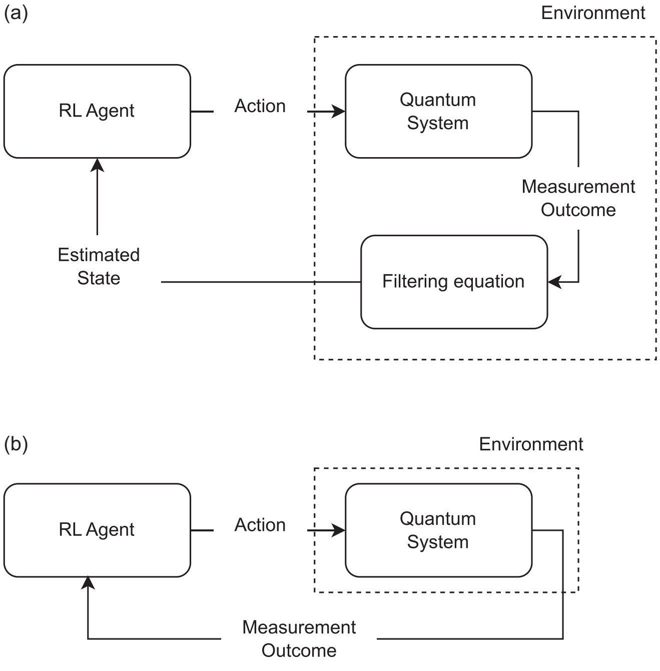

FIG. 2. (a) The first feedback scenario in which the environment (square outlined with dashed lines) outputs the estimated state. (b) The second feedback scenario in which the environment (square outlined with dashed lines) provides to the agent just the outcomes of the measurements performed on the quantum system.

the results of quantum measurements. Our approach consists of training a RL agent able to learn adaptive strategies through trial and error, iteratively refining its control policies based on the outcomes of quantum measurements or on an estimated state of the quantum system. The key elements of the RL setting are described in the following.

1. RL environment

The RL environment in this context is associated with a quantum system undergoing potential noisy evolution, coupled with a measurement apparatus. The best available description of the environment is quantum, where the state is described by density operators and may be subject to probabilistic changes as described in Sec. II. The environment takes as input the control parameter β and outputs the outcome of the measurement or, when available, the updated estimation of the state computed through a filtering equation (Fig. 2). The selection of the environment output depends on the training model.

2. Agent task

In the context of the state preparation problem, the objective is to identify the optimal control law that determines β based on the measurement outcome and, when accessible, on the estimated state, to drive the quantum state as close as possible to a target state. This is necessary to optimize the policy π_θ that dictates the selection of β. We want to remark that the representation of the state plays a key role in the optimization of the policy, as important as the design of the reward function.

3. Reward function

The choice of the reward function is a key element toward the optimization of the agent's policy. Our work addresses two main scenarios: in the first the RL agent is supplied with an estimated density operator, while in the second the agent receives only measurement outcomes. In the former case, having access to the system's density operator, we naturally opt for fidelity as the reward function, quantifying the proximity between the estimated state and the target state. In the latter scenario, characterized by a less informative environment, our strategy involves assigning a positive reward when the measurement outcome aligns with our target state and a negative reward otherwise.

V. CONTROL AND LEARNING SCENARIOS

A. Nominal and filtering dynamics

Before delving deeper into the control scenarios and, in particular, RL scenarios, we need to define two additional states and their evolution: the *nominal state* and the *filtered state*. The need for considering such states emerges because, while our RL agent is assumed to have full information about some key quantities entering the model [depending on the particular setup, these might include the initial state of the system (ρ_0, ρ_t), the control parameters β_t, and the measurement operator and outcomes $M_{l,\epsilon}$ and l_t], we suppose that it does not account for the noise present in the actual system dynamics. This will allow us to test the robustness to these strategies derived under ideal conditions to the introduction of noise.

In light of this, we define the nominal state $\bar{\rho}$ as the solution to the following *nominal dynamics*:

$$\begin{cases} \bar{\rho}(t+1) = \mathcal{M}^{\epsilon}_{\bar{l}(t)} \circ \mathcal{U}_{\beta_t}[\bar{\rho}(t)] \\ \bar{\rho}(0) = \rho_0, \end{cases} \tag{21}$$

where β_t represents the actual control input fed to the system, and $\bar{l}(t)$ indicates the measurement outcome without noise in the system dynamics, artificially sampled from a distribution with probabilities:

$$p[\bar{l}(t)] = \mathrm{tr}\{M^{\dagger}_{l_t} M_{l_t}[\bar{\rho}(t)]\}. \tag{22}$$

Next, we define the evolution for the filtered state $\hat{\rho}(t)$, where we consider the real measurement outcomes, but evolve the state using only the nominal, noiseless dynamics. This filtered state is then the solution of the *filtering dynamics*:

$$\begin{cases} \hat{\rho}(t+1) = \mathcal{M}^{\epsilon}_{l(t)} \circ \mathcal{U}_{\beta_t}[\hat{\rho}(t)] \\ \hat{\rho}(0) = \rho_0. \end{cases} \tag{23}$$

It is worth highlighting that the difference with respect to the previous equation lays in the different origin and distribution of the outcomes: for the filtered state we consider the *actual* outcomes $l(t)$, with associated distribution depending on the presence of noise and the evolution of the true state of the system $\rho(t)$, which evolves as in (5):

$$p[l(t)] = \mathrm{tr}\{M^{\dagger}_{l_t} M_{l_t} \mathcal{N}_{\alpha}[\rho(t)]\}. \tag{24}$$

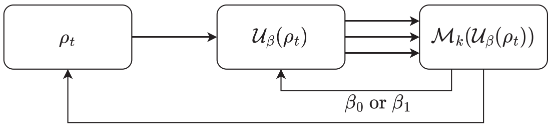
ρ_t
$\mathcal{U}_\beta(\rho_t)$
$\mathcal{M}_k(\mathcal{U}_\beta(\rho_t))$
β_0 or β_1

FIG. 3. Basic controller block scheme.

B. Basic controller

One possible control policy to stabilize the system described in the previous section is the use of a simple feedback law derived from the nominal system with perfect (projective) measurements, that is, when $\alpha = \epsilon = 0$. The main idea is to select the control parameter according to the outcome of the most recent measurement, maximizing the probability of transition toward the target (Fig. 3). To do so we call β_0 and β_1 the control parameter to be applied when the outcomes are, respectively, $l = 0$ and $l = 1$, while we pose $\beta_2 = 0$ since it corresponds, at least in this setting, to the target. These two fixed parameters are computed solving the following optimization problem:

$$\begin{aligned}\beta_0 &= \arg\max_\beta \mathrm{tr}[|2\rangle\langle 2|\mathcal{U}_\beta(|0\rangle\langle 0|)]\\ &= \arg\max_\beta \langle 2|\mathcal{U}_\beta(|0\rangle\langle 0|)|2\rangle, \qquad (25)\\ \beta_1 &= \arg\max_\beta \mathrm{tr}[|2\rangle\langle 2|\mathcal{U}_\beta(|1\rangle\langle 1|)]\\ &= \arg\max_\beta \langle 2|\mathcal{U}_\beta(|1\rangle\langle 1|)|2\rangle. \qquad (26)\end{aligned}$$

The values of the two parameters that maximize the previous figure of merit are $\beta_0 = \beta_1 = 1$.

C. Model-based learning scenario

In the model-based scenario (MBS), we consider a quantum system with a known initial state represented by the density operator $\bar{\rho}_0$. In this scenario, we assume the agent has access to the system's state at time t, the controls β_t, and the measurement set $\{M_{\bar{l}}^{\epsilon}\}$. During the training phase, the agent is training on the evolution and measurements statistics associated with the *nominal dynamics* (21). Indeed, in this phase the noise operator, denoted by $\mathcal{N}_\alpha$, is not included in the model ($\alpha = 0$), and the distribution of the measurement outcomes is the one provided in Eq. (22).

In this phase, the agent receives the current state of the quantum system $\bar{\rho}_t$ as input at each time step t and outputs the control parameter β (Fig. 4). The agent learns to make decisions about β based on the observed state and the ultimate goal, which consists of maximization of the fidelity function as a figure of merit.

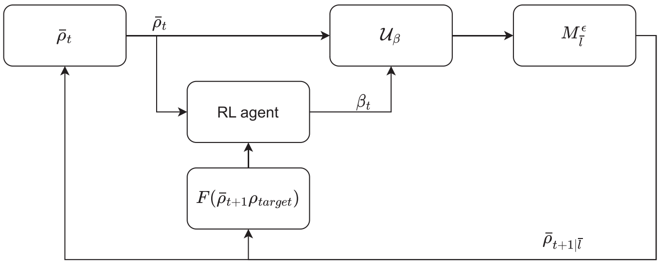

FIG. 4. Training scheme of the model-based scenario.

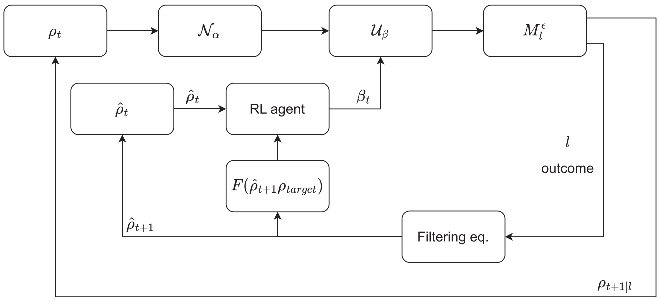

FIG. 5. Training scheme of the data-based scenario.

During the validation phase, noise is injected in the system ($\mathcal{N}_\alpha$ with $\alpha \neq 0$), so that the dynamic of the system follows Eq. (5). The agent has to choose the control parameter β at every time step t, but this time the dynamics of the gym environment follow the *filtered equation* (23).

D. Data-based learning scenario

In the data-based scenario (DBS), we consider a quantum system with a known initial state represented by the density operator $\hat{\rho}_0$. As for the one before, the agent is aware of the system's state at time t, the control parameter β, and the measurement set M. During the training phase, the dynamics of the gym environment where the agent is trained is represented by the *filtered state*. When we get a measurement outcome, which in this case is sampled from the real distribution [Eq. (24)], we compute the current state with the best estimation possible, which is given from the following filtering equation:

$$\hat{\rho}_{t+1} = \frac{M_{\hat{l}_t}^{\epsilon} U_\beta \hat{\rho}_t U_\beta^\dagger M_{\hat{l}_t}^{\epsilon\dagger}}{\mathrm{tr}(\cdot)}. \qquad (27)$$

In the training phase we fed the agent with the estimate of the current state ρ_t, and we collect as output the control parameter $\beta(t)$ (Fig. 5). Also, in this scenario the reward function the agent tries to maximize is the fidelity, which is computed as in Eq. (19).

During the validation phase we keep the same dynamics as in the training phase. Also, this time the agent has to select at every time step t the best control parameter β to stabilize the state.

E. Model-free scenario: Quantum observable Markov decision process

In the model-free scenario we train the controller without any reference model, quantum or classical. The approach will be called quantum observable Markov decision processes (QOMDPs), as it is inspired by the works of Refs. [27,29]. We consider a quantum system characterized by an initial state ρ_0 that evolves over time according to its dynamics, which depends on a noise parameter α [as specified in Eq. (5)]. In this scenario, the distribution of measurement outcomes follows the true distribution of outcomes, similar to the DBS approach.

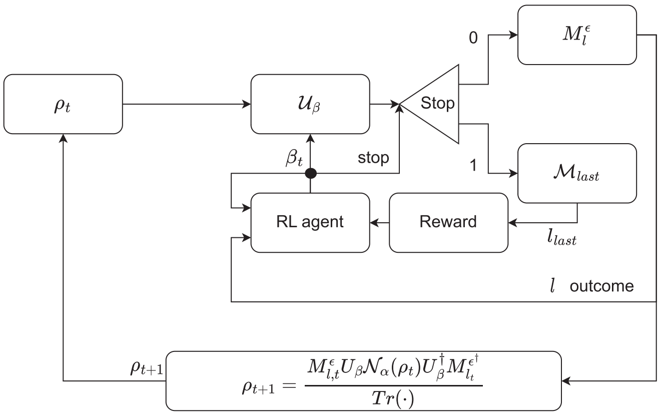

FIG. 6. Training scheme of the QOMDP scenario.

The agent in this QOMDP framework does not receive the state of the quantum system ρ_t or any estimation thereof, as is the cases of MBS or DBS. It receives only two pieces of information at each time step t: the outcome of the last measurement, denoted as l_{t-1}, and the previous control action, denoted as β_{t-1}.

At each time step t, the agent has to make two decisions: selecting the control action β_t and specifying the value of the *stop action*. If the stop action equals 1, the episode ends; otherwise, if it takes the value 0, the episode continues. At the initial time step $t = 0$, no action is performed, i.e., $\beta = 0$ and stop $= 0$. This leads to a unitary evolution represented by the identity operator $U(0) = I_{n\times n}$. After this, the first state provided to the agent is always in the form of the column vector $[l_{t=0}, 0]^T$.

To set up the training environment, we introduce a new set of measurements called the *last observation*, denoted as l_{last} (Fig. 6). This observation is obtained by measuring the quantum system using a projective set of measurement operators $M_0^{\text{end}}, M_1^{\text{end}}, M_2^{\text{end}}$ as follows:

$$M_0^{\text{end}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$M_1^{\text{end}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$M_2^{\text{end}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{28}$$

Once the last observation l_{last} is measured, we can compute the reward function as follows:

$$R(\text{stop}, \text{l}_{\text{last}}, \text{done}) = \begin{cases} 0, & \text{if stop} = 0 \text{ done} = \text{false} \\ -1, & \text{if stop} = 0 \text{ done} = \text{true} \\ r_{\text{fun}}(l_{\text{last}}), & \text{if stop} = 1 \end{cases}, \tag{29}$$

where r_{fun} is defined as

$$r_{\text{fun}}(l_{\text{last}}) = \begin{cases} +1, & \text{if } l_{\text{last}} = l_{\text{target}} \\ -1, & \text{otherwise} \end{cases}. \tag{30}$$

TABLE I. RL models summary.

Model	Training	Train α	Train ϵ	Validation
MBs	Nominal dynamics	0	Every ϵ	Filtered dynamics
DBs	Filtered dynamics	Every α	Every ϵ	Filtered dynamics
QOMDP	Nominal data	0	Every ϵ	Real data

In this expression, l_{target} represents a target observation value for the quantum state we want to reach.

During the validation phase, the quantum system's dynamics remain unchanged. The reward function is no longer utilized or considered, since it is part of the learning process. The primary focus is on evaluating the agent's performance and decision-making without incorporating the reward function. The agent's actions are assessed solely based on their impact on the quantum system's evolution and fidelity with respect to the target state.

VI. NUMERICAL ANALYSIS

A. Experiments setup

In this section of the paper, we report on the results of the simulations [62] in order to assess the performance of various RL models with respect to variation in the measurement accuracy and the noise level. We summarize in the following table the main peculiarities of each model (Table I). Our comparative analysis focused on evaluating the fidelity of the state $\rho(t)$, which was generated by applying the true dynamics while accounting for noise and utilizing control parameters determined by the RL agent or by the basic controller. We considered the three types of noise that we had previously introduced: the random permutation noise, the depolarizing channel, and the amplitude damping channel.

We trained and subsequently tested the various RL models for all possible configurations of the following parameters [63].

To ensure a robust evaluation of our simulations, we adopted a standardized approach. Specifically, for each simulation, we gathered a dataset consisting of 1000 samples. Subsequently, we computed both the mean and the standard deviation from this dataset, visually represented by lines and shaded regions. respectively, in the plots.

It is noteworthy that our modeling approaches, MBS and DBS, were trained using the PPO algorithm. For these methods, we employed a conventional multilayer perceptron (MLP) network architecture. In contrast, the QOMDP approach underwent training using PPO in conjunction with a long short-term memory (LSTM) network. This choice was made to enable the tracking of previous measurement outcomes.

B. Results

In the first plot (Fig. 7) we display the highest intensity of noise that an agent can handle while maintaining a final fidelity of at least 0.9, for each ϵ. Some observations follow from the comparison of Fig. 7. Firstly, when dealing with

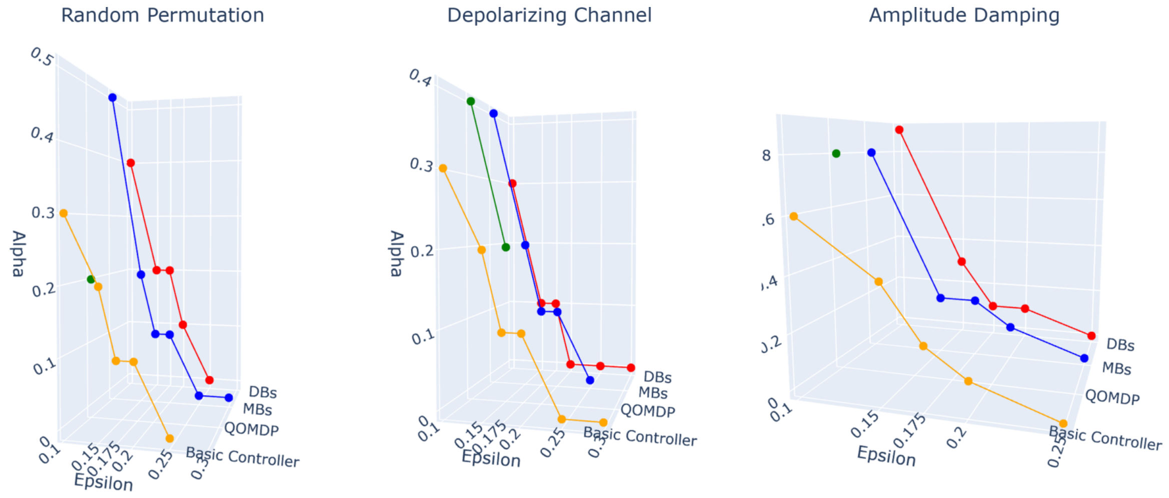

FIG. 7. The highest value of noise intensity α that can be handled by each controller while maintaining a fidelity of at least 0.9. Missing points in the curve indicate that the target fidelity cannot be guaranteed.

precise measurements (i.e., with $\epsilon = 0.1$), it is evident that the RL agents exhibit superior performance compared to the basic controller.

Notably, when confronted with random permutation noise, both the DBS and MBS agents demonstrate their robustness by effectively managing the situation up to $\alpha = 0.5$. Similarly,

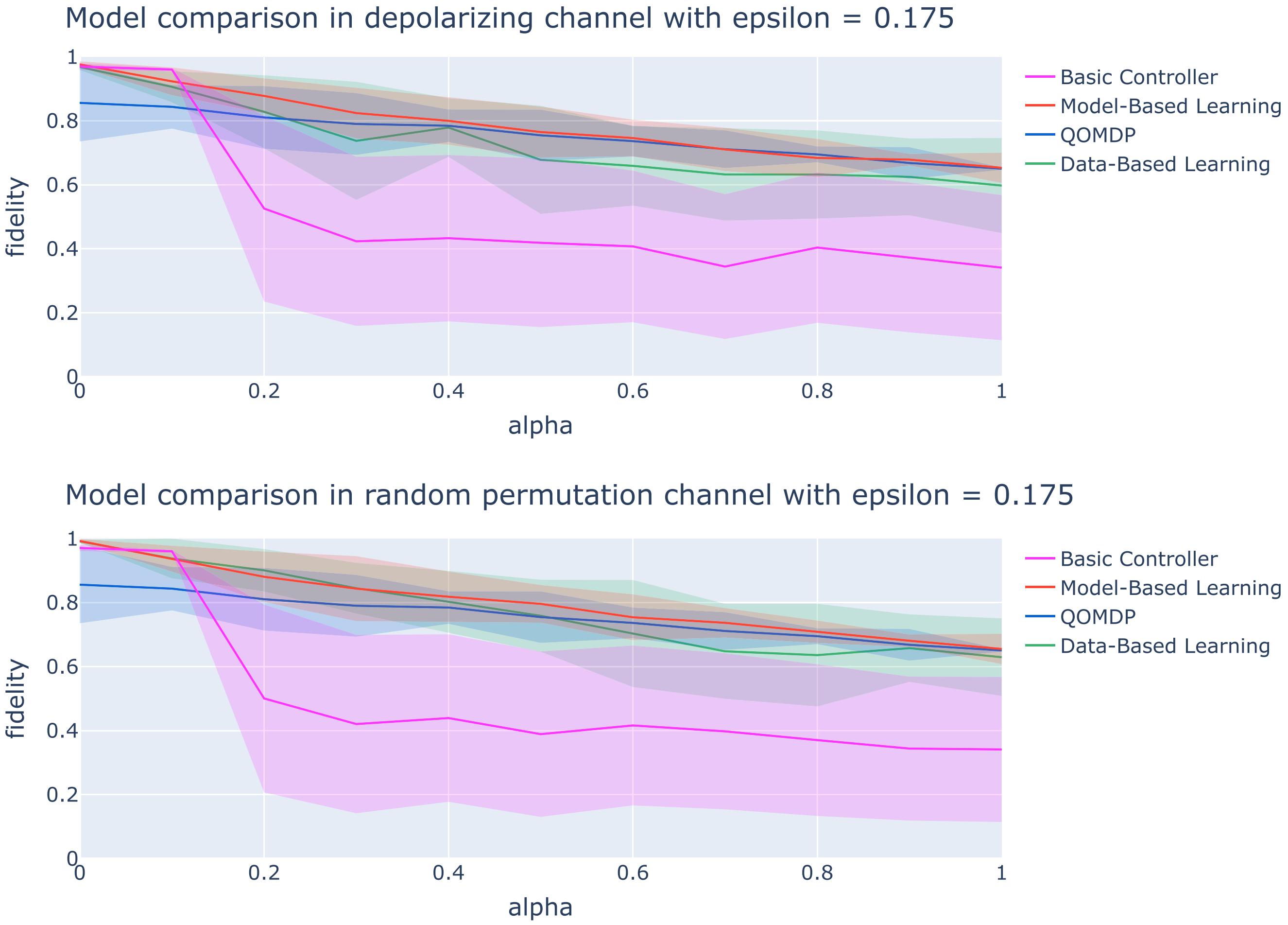

FIG. 8. Simulation results for $\epsilon = 0.175$ are depicted. (Top) Behavior of various agents concerning the depolarizing channel and (bottom) their performance with respect to the random permutation channel. A discernible trend emerges wherein RL agents exhibit a gradual decline in performance, whereas the deterministic controller shows a stark deterioration beyond a certain α value.

when faced with the challenges of introducing the depolarizing channel noise at $\epsilon = 0.1$, the MBS scenario outperforms other approaches, extending its capability to handle noise levels up to $\alpha = 0.4$. For the amplitude damping channel, it is evident across all considered scenarios that controllers exhibit a higher noise threshold compared to the other two noise models—see Fig. 7 for high measurement accuracy ($\epsilon \leqslant 0.15$). Notably, the DBS agent demonstrates superior performance, achieving target fidelity value with noise intensity $\alpha = 0.9$. Following closely are both the MBS and QOMDP approaches, attaining similar fidelity values till $\alpha = 0.8$. The deterministic approach remains limited, achieving target fidelity for $\alpha = 0.6$. For $\epsilon > 0.1$, MBS, DBS, and the deterministic controller exhibit comparable performances. On the other hand, the performance of the QOMDP approach degrades quickly, and does not reach target fidelity $\epsilon > 0.1$.

It is worth observing that in general the QOMDP agent displays commendable performance as long as measurements are informative. This outcome can be attributed to the reliance of the QOMDP agent on the quality of the latter, which plays a pivotal role in determining when to execute control actions effectively. On the other hand, it is not able to guarantee target fidelity as soon as the measurement quality decreases. The simulations also confirm the presence of a *trade-off between measurement accuracy and the level of noise* that all the controllers can effectively cope with.

To further elucidate this phenomenon, we present two paradigmatic plots illustrating the performance of the various methods at various noise levels when $\epsilon = 0.175$. These two plots offer some critical insights: (1) while there is an initial resemblance in behavior between the basic controller and the MBS and DBS agents when confronted with low noise intensity, as the noise intensity escalates, the performance of the basic controller drops; and (2) the performance of the QOMDP is lower than the other RL methods.

It is important to note that we do not contend that the fidelity achieved by the RL agents is necessarily optimal. Instead, our contention is that RL presents a more robust control approach compared to the basic controller. In other words, it excels in maintaining control performance as noise levels increase.

Last, it is possible to note that the number of necessary steps in order to achieve a fidelity greater than or equal to 0.9 in general is lower for the RL agents compared to the basic controller. This result is summarized in Fig. 9. Summarizing the findings of the extended simulations we ran, we can conclude that:

(1) For low noise intensity, the basic controller and the RL controllers have similar performance;

(2) It is enough, however, to raise the measurement inaccuracy ϵ to 0.1 to notice a significant advantage of the RL controllers in terms of the amount of noise they can withstand while maintaining the fidelity threshold (Fig. 8);

(3) The model-based controller competes with the data-based one;

(4) The trade-off between measurement accuracy and noise is evident, and becomes less taxing for RL-based controllers;

(5) The RL controller tends to converge to high fidelity in less steps than the basic controller.

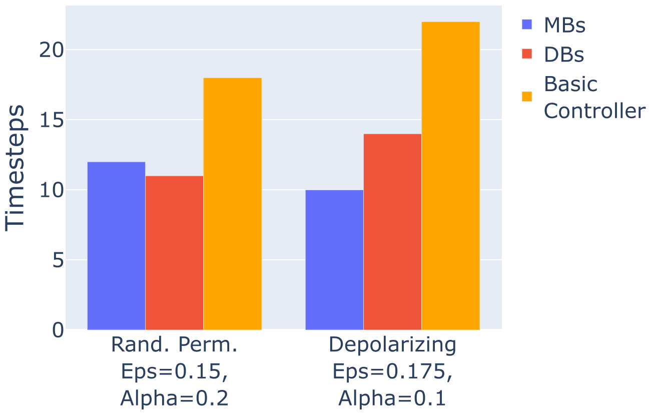

FIG. 9. Examples indicating the necessary number of time steps required to achieve a fidelity of $\geqslant 0.9$. It is evident that RL agents outperform the basic controller in terms of speed in both scenarios.

VII. CONCLUSIONS

In this study, we conducted an assessment of the performance of various RL models in controlling quantum systems subject to noise and uncertainties toward a state preparation task. Our performance analysis focused on the fidelity of a quantum state $\rho(t)$ with respect to a target pure state, which was generated by applying true dynamics while accounting for noise and utilizing control parameters determined by RL agents or a basic controller.

The study investigated a range of RL models, including MBS, DBS, and QOMDP. To empower these models, we utilized the PPO algorithm and distinct network architectures.

In the MBS and DBS the agents harnessed the PPO algorithm, paired with a traditional MLP network architecture. Their training phases were based on data obtained from the nominal dynamics [as defined in Eq. (21)] for the MBS, and real data fed to the filtering dynamics [as defined in Eq. (23)] equations for the DBS.

In contrast to MBS and DBS, the QOMDP agent employed PPO in conjunction with a LSTM network architecture. This approach allowed the QOMDP agent to track and incorporate previous measurement outcomes into its decision-making process. Importantly, the QOMDP agent did not construct an updated system state; instead, its actions were solely informed by measurement outcomes.

We considered a testbed system on which running the simulations in order to compare the behavior of the different control scenarios. We tested the performance of each model with respect to both the depolarizing channel and a random permutation channel accounting for different noise intensity and measurements inaccuracy parameters (Table II).

To establish a benchmark for the aforementioned models, we introduced a reference control strategy, referred to as the

TABLE II. Training and test parameters.

Parameter	Values										
α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
ϵ	0.1	0.15	0.175	0.2	0.25	0.3					

basic controller. This basic controller is designed to obtain perfect state preparation when both uncertainty in measurements and noise intensity parameters, ϵ and α, are set to zero, maximizing the probability of transitioning from the quantum states $|0\rangle\langle 0|$ and $|1\rangle\langle 1|$ to the target state $|2\rangle\langle 2|$.

The numerical analysis we performed analyzed the performance of various controllers in the presence of noise and measurement inaccuracy. We here make an attempt of distilling a few key messages and physical insight on the behavior of RL controllers. In scenarios with low noise intensity, both the basic controller and RL controllers demonstrate, as expected, comparable performance. This implies that traditional control methods can be as effective as reinforcement learning-based controllers in relatively noise-free environments. However, when we increase the measurement inaccuracy to a modest value of $\epsilon = 0.1$, the RL controllers exhibit a clear advantage. They excel in withstanding noise while still maintaining the desired fidelity threshold (greater than or equal to 0.9). This showcases the robustness and adaptability of RL-based control in challenging conditions.

Our analysis also highlights a clear trade-off between measurement accuracy and noise tolerance for all methods, which is, however, less constraining for RL-based controllers. The latter appear to be better suited to navigate this delicate balance.

Moreover, RL controllers require fewer steps to reach a high level of fidelity compared to the basic controller. This suggests that RL methods can offer faster and more efficient control solutions in certain scenarios.

Lastly, one of the most remarkable observations is that such superior robustness with respect to noise, i.e., unmodeled Markovian dynamics on the system, is obtained *even if the controller is trained with data coming from a simulation that does not include noise.* This opens the possibility of using RL techniques in practical scenarios, where the amount of data needed to obtain the controller would entail a prohibitively long time to be obtained. In fact, even for classical systems, these issues can represent critical hurdles to the successful employments of RL for real world systems.

In summary, our study underscores the adaptability and resilience of reinforcement learning controllers in the face of noise and measurement inaccuracy. While traditional control methods are effective in low noise conditions, RL controllers show a clear advantage when confronted with more challenging and uncertain environments. The choice between these approaches should be made based on the specific requirements and conditions of the task at hand.

APPENDIX: NETWORK ARCHITECTURES AND TRAINING PARAMETERS

This Appendix provides a detailed description of the network architecture and hyperparameters employed in the experiments. An understanding of these details is crucial for replicating and extending the presented work.

1. MBS and DBS scenarios

The actor-critic policy networks employed in the MBS and DBS scenarios consist of three main components:

(1) Flatten extractor: This component flattens the input state vector, converting it into a one-dimensional representation.

(2) Policy and value function extractors: These extractors further process the flattened state representation using two separate MLP architectures, one for policy estimation and the other for value function approximation. Each MLP consists of three hidden layers with 64 nodes each, using Tanh activation functions.

(3) Action and value nets: These nets receive the output of the respective extractors and generate the policy distribution and value function estimate, respectively. The action net is a single-layer linear transformation with one output node, representing the predicted action probabilities. The value net also utilizes a single-layer linear transformation to output a scalar representing the predicted state value.

The trainings of the networks for these two scenarios were conducted with the following parameters:

(1) Batch size: The batch size for each training update was set to 512. This value represents the number of state-action pairs used to update the network parameters.

(2) Number of steps per update: For each training update, the agent interacts with the environment for a total of 512 steps. This corresponds to the number of state-action pairs sampled before updating the policy and value function parameters.

(3) Learning rate: The learning rate for the optimizer was set to 1×10^{-4}. This value controls the step size during gradient descent updates, ensuring a balance between exploration and exploitation.

For all the other parameters we used the default Stable-Baselines3 settings.

2. QOMDP scenario

The LSTM policy network employed in this scenario consists of four main components:

(1) Flatten extractor: This component flattens the input state vector, converting it into a one-dimensional representation.

(2) Policy and value function extractors: These extractors further process the flattened state representation using two separate MLP architectures, one for policy estimation and the other for value function approximation. Each MLP consists of three hidden layers with 64 nodes each, using Tanh activation functions.

(3) Action and value nets: These nets receive the output of the respective extractors and generate the policy distribution and value function estimate, respectively. The action net is a single-layer linear transformation with two output nodes, representing the predicted action probabilities for the two possible actions. The value net also utilizes a single-layer linear transformation to output a scalar representing the predicted state value.

(4) Recurrent architecture: To capture temporal dependencies in the environment, the network utilizes two LSTM units, one for policy and one for value function estimation. These LSTM units process the sequences of state and action inputs, enabling the network to learn long-range dependencies and adapt its behavior accordingly.

The training of the recurrent actor-Ccritic policy network was conducted with the following parameters:

(1) Batch size: The batch size for each training update was set to 512. This value represents the number of state-action pairs used to update the network parameters.

(2) Number of steps per update: For each training update, the agent interacts with the environment for a total of 512 steps. This corresponds to the number of state-action pairs sampled before updating the policy and value function parameters.

(3) Learning rate: The learning rate for the optimizer was set to 3×10^{-4}. This value controls the step size during gradient descent updates, ensuring a balance between exploration and exploitation.

For all the other parameters we used the default Stable-Baselines3 settings.

[1] C. Altafini and F. Ticozzi, Modeling and control of quantum systems: An introduction, IEEE Trans. Autom. Control **57**, 1898 (2012).

[2] D. d'Alessandro, *Introduction to Quantum Control and Dynamics* (CRC Press, Boca Raton, 2021).

[3] C. P. Koch, U. Boscain, T. Calarco, G. Dirr, S. Filipp, S. J. Glaser, R. Kosloff, S. Montangero, T. Schulte-Herbrüggen, D. Sugny, and F. K. Wilhelm, Quantum optimal control in quantum technologies. Strategic report on current status, visions and goals for research in Europe, EPJ Quantum Technol. **9**, 19 (2022).

[4] D. Dong and I. R. Petersen, Quantum control theory and applications: A survey, IET Control Theory Appl. **4**, 2651 (2010).

[5] J. C. Doyle, B. A. Francis, and A. R. Tannenbaum, *Feedback Control Theory* (Courier Corporation, North Chelmsford, 2013).

[6] V. P. Belavkin, Quantum stochastic calculus and quantum nonlinear filtering, J. Multivariate Anal. **42**, 171 (1992).

[7] V. P. Belavkin, Towards the theory of control in observable quantum systems, arXiv:quant-ph/0408003.

[8] H. M. Wiseman and G. J. Milburn, Quantum theory of optical feedback via homodyne detection, Phys. Rev. Lett. **70**, 548 (1993).

[9] H. M. Wiseman and G. J. Milburn, *Quantum Measurement and Control* (Cambridge University Press, Cambridge, 2009).

[10] A. Barchielli and M. Gregoratti, *Quantum Trajectories and Measurements in Continuous Time: the Diffusive Case* (Springer, New York, 2009), Vol. 782.

[11] M. Mirrahimi and R. Van Handel, Stabilizing feedback controls for quantum systems, SIAM J. Control Optim. **46**, 445 (2007).

[12] M. Yanagisawa and H. Kimura, Transfer function approach to quantum control-part I: Dynamics of quantum feedback systems, IEEE Trans. Autom. Control **48**, 2107 (2003).

[13] L. Bouten, R. Van Handel, and M. R. James, A discrete invitation to quantum filtering and feedback control, SIAM Rev. **51**, 239 (2009).

[14] J. Gough and M. R. James, The series product and its application to quantum feedforward and feedback networks, IEEE Trans. Autom. Control **54**, 2530 (2009).

[15] J. Zhang, Y.-X. Liu, R.-B. Wu, K. Jacobs, and F. Nori, Quantum feedback: Theory, experiments, and applications, Phys. Rep. **679**, 1 (2017).

[16] H. I. Nurdin, M. R. James, and I. R. Petersen, Coherent quantum LGQ control, Automatica **45**, 1837 (2009).

[17] F. Ticozzi and L. Viola, Analysis and synthesis of attractive quantum Markovian dynamics, Automatica **45**, 2002 (2009).

[18] A. C. Doherty, S. Habib, K. Jacobs, H. Mabuchi, and S. M. Tan, Quantum feedback control and classical control theory, Phys. Rev. A **62**, 012105 (2000).

[19] W. Liang, T. Grigoletto, and F. Ticozzi, Dissipative feedback switching for quantum stabilization, Automatica **165**, 111659 (2024).

[20] C. Sayrin, I. Dotsenko, X. Zhou, B. Peaudecerf, T. Rybarczyk, S. Gleyzes, P. Rouchon, M. Mirrahimi, H. Amini, M. Brune, J.-M. Raimond, and S. Haroche, Real-time quantum feedback prepares and stabilizes photon number states, Nature (London) **477**, 73 (2011).

[21] F. Ticozzi, A. Ferrante, and M. Pavon, Robust steering of n-level quantum systems, IEEE Trans. Autom. Control **49**, 1742 (2004).

[22] R. Van Handel, Filtering, stability, and robustness, Ph.D. thesis, California Institute of Technology (2007).

[23] I. R. Petersen, V. Ugrinovskii, and M. R. James, Robust stability of quantum systems with a nonlinear coupling operator, in *Proceedings of the 2012 IEEE 51st IEEE Conference on Decision and Control (CDC)* (IEEE, New York, 2012), pp. 1078–1082.

[24] W. Liang, N. H. Amini, and P. Mason, Robust feedback stabilization of *N*-level quantum spin systems, SIAM J. Control Optim. **59**, 669 (2021).

[25] S. S. Ge, T. L. Vu, and T. H. Lee, Quantum measurement-based feedback control: A nonsmooth time delay control approach, SIAM J. Control Opt. **50**, 845 (2012).

[26] W. Liang, K. Ohki, and F. Ticozzi, Exploring the robustness of stabilizing controls for stochastic quantum evolutions, arXiv:2311.04428.

[27] V. V. Sivak, A. Eickbusch, H. Liu, B. Royer, I. Tsioutsios, and M. H. Devoret, Model-free quantum control with reinforcement learning, Phys. Rev. X **12**, 011059 (2022).

[28] R. Porotti, A. Essig, B. Huard, and F. Marquardt, Deep reinforcement learning for quantum state preparation with weak nonlinear measurements, Quantum **6**, 747 (2022).

[29] J. Barry, D. T. Barry, and S. Aaronson, Quantum partially observable Markov decision processes, Phys. Rev. A **90**, 032311 (2014).

[30] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, Stable-baselines3: Reliable reinforcement learning implementations, J. Mach. Learn. Res. **22**, 1 (2021).

[31] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, Reinforcement learning in different phases of quantum control, Phys. Rev. X **8**, 031086 (2018).

[32] 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. **5**, 85 (2019).

[33] S.-F. Guo, F. Chen, Q. Liu, M. Xue, J.-J. Chen, J.-H. Cao, T.-W. Mao, M. K. Tey, and L. You, Faster state preparation across quantum phase transition assisted by reinforcement learning, Phys. Rev. Lett. **126**, 060401 (2021).

[34] J. Mackeprang, D. B. R. Dasari, and J. Wrachtrup, A reinforcement learning approach for quantum state engineering, Quantum Mach. Intell. **2**, 5 (2020).

[35] S. Z. Baba, N. Yoshioka, Y. Ashida, and T. Sagawa, Deep reinforcement learning for preparation of thermal and prethermal quantum states, Phys. Rev. Appl. **19**, 014068 (2023).

[36] F. Preti, M. Schilling, S. Jerbi, L. M. Trenkwalder, H. P. Nautrup, F. Motzoi, and H. J. Briegel, Hybrid discrete-continuous compilation of trapped-ion quantum circuits with deep reinforcement learning Quantum **8**, 1343 (2024).

[37] 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 Networks Learn. Syst. **25**, 920 (2013).

[38] R. Porotti, D. Tamascelli, M. Restelli, and E. Prati, Coherent transport of quantum states by deep reinforcement learning, Commun. Phys. **2**, 61 (2019).

[39] T. Haug, W.-K. Mok, J.-B. You, W. Zhang, C. Eng Png, and L.-C. Kwek, Classifying global state preparation via deep reinforcement learning, Mach. Learn.: Sci. Technol. **2**, 01LT02 (2021).

[40] M. Bukov, Reinforcement learning for autonomous preparation of Floquet-engineered states: Inverting the quantum Kapitza oscillator, Phys. Rev. B **98**, 224305 (2018).

[41] 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. **127**, 190403 (2021).

[42] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, Reinforcement learning with neural networks for quantum feedback, Phys. Rev. X **8**, 031084 (2018).

[43] 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. **8**, 29 (2021).

[44] M. Li, T. W. Hänsch, and R. Blatt, Realizing a deep reinforcement learning agent for real-time quantum feedback, Nature (London) **551**, 378 (2017).

[45] N. Meyer, C. Ufrecht, M. Periyasamy, D. D. Scherer, A. Plinge, and C. Mutschler, A survey on quantum reinforcement learning, arXiv:2211.03464.

[46] K. Reuer, J. Landgraf, T. Fösel, J. O'Sullivan, L. Beltrán, A. Akin, G. J. Norris, A. Remm, M. Kerschbaum, J.-C. Besse, F. Marquardt, A. Wallraff, and C. Eichler, Realizing a deep reinforcement learning agent for real-time quantum feedback, Nat. Commun. **14**, 7138 (2023).

[47] H.-Y. Chen, Y.-J. Chang, S.-W. Liao, and C.-R. Chang, Deep-Q learning with hybrid quantum neural network on solving maze problems, arXiv:2304.10159.

[48] A. Bolens and M. Heyl, Reinforcement learning for digital quantum simulation, Phys. Rev. Lett. **127**, 110502 (2021).

[49] S. Wu, S. Jin, D. Wen, D. Han, and X. Wang, Quantum reinforcement learning in continuous action space, arXiv:2012.10711.

[50] S. Y.-C. Chen, C.-H. H. Yang, J. Qi, P.-Y. Chen, X. Ma, and H.-S. Goan, Variational quantum circuits for deep reinforcement learning, IEEE Access **8**, 141007 (2020).

[51] V. V. Sivak, A. Eickbusch, B. Royer, S. Singh, I. Tsioutsios, S. Ganjam, A. Miano, B. L. Brock, A. Z. Ding, L. Frunzio, S. M. Girvin, R. J. Schoelkopf, and M. H. Devoret, Real-time quantum error correction beyond break-even, Nature (London) **616**, 50 (2023).

[52] P. Andreasson, J. Johansson, S. Liljestrand, and M. Granath, Quantum error correction for the toric code using deep reinforcement learning, Quantum **3**, 183 (2019).

[53] T. Fösel, M. Y. Niu, F. Marquardt, and L. Li, Quantum circuit optimization with deep reinforcement learning, arXiv:2103.07585.

[54] E. S. Matekole, E. Ye, R. Iyer, and S. Y.-C. Chen, Decoding surface codes with deep reinforcement learning and probabilistic policy reuse, arXiv:2212.11890.

[55] H. P. Nautrup, N. Delfosse, V. Dunjko, H. J. Briegel, and N. Friis, Optimizing quantum error correction codes with reinforcement learning, Quantum **3**, 215 (2019).

[56] F. Ticozzi, R. Lucchese, P. Cappellaro, and L. Viola, Hamiltonian control of quantum dynamical semigroups: Stabilization and convergence speed, IEEE Trans. Autom. Control **57**, 1931 (2012).

[57] A. Essig, Q. Ficheux, T. Peronnin, N. Cottet, R. Lescanne, A. Sarlette, P. Rouchon, Z. Leghtas, and B. Huard, Multiplexed photon number measurement, Phys. Rev. X **11**, 031045 (2021).

[58] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).

[59] S. Bolognani and F. Ticozzi, Engineering stable discrete-time quantum dynamics via a canonical QR decomposition, IEEE Trans. Autom. Control **55**, 2721 (2010).

[60] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. (MIT Press, Cambridge, 2018).

[61] We here use the standard RL nomenclature, but note that this is not necessarily a (quantum) state. Similarly, in a RL context, the environment is actually the system to be controlled, see Sec. IV C for more details.

[62] The primary research efforts were conducted at the University of Padova, while certain simulations concerning DBs and MBs were performed at FZJ.

[63] The code is available at https://github.com/ManuelGuatto/RL_4_Robust_QC.git.