

Quantum-enhanced deliberation of learning agents using trapped ions

Vedran Dunjko,^{1,2,3,*} Nicolai Friis,^{1,†} and Hans J. Briegel^{1,2,‡}

¹*Institute for Quantum Optics and Quantum Information,
Austrian Academy of Sciences, Technikerstraße 21a, A-6020 Innsbruck, Austria*

²*Institute for Theoretical Physics, University of Innsbruck, Technikerstraße 25, A-6020 Innsbruck, Austria*

³*Laboratory of Evolutionary Genetics, Division of Molecular Biology,
Ruđer Bošković Institute, Bijenička cesta 54, HR-10000 Zagreb, Croatia*

(Dated: July 11, 2014)

A scheme that successfully employs quantum mechanics in the design of autonomous learning agents has recently been reported in the context of the projective simulation (PS) model for artificial intelligence. In that approach, the key feature of a PS agent, a specific type of memory which is explored via random walks, was shown to be amenable to quantization. In particular, classical random walks were substituted by Szegegy-type quantum walks, allowing for a speed-up. In this work we propose how such classical and quantum agents can be implemented in systems of trapped ions. We employ a generic construction by which the classical agents are ‘upgraded’ to their quantum counterparts by nested coherent controlization, and we outline how this construction can be realized in ion traps. Our results provide a flexible modular architecture for the design of PS agents. Furthermore, we present numerical simulations of simple PS agents which analyze the robustness of our proposal under certain noise models.

PACS numbers: 07.05.Mh, 03.67.Lx, 37.10.Ty, 05.40.Fb

I. INTRODUCTION

In the past decades, quantum physics has been employed to enhance communication and information processing with significant success, laying the foundation for the now well established fields of quantum computation and quantum information [1–5]. In contrast, the potential of merging the related, but distinct, field of artificial intelligence (AI) with quantum physics is significantly less well-understood. Thus far, advances in this field have been reported mostly for algorithmic approaches to applied AI-related tasks, e.g., (un-)supervised data clustering and process replication, where selected quantum algorithms could be utilized [6–10].

On the other hand, the first result showing that quantum mechanics can also aid in the complementary task of designing autonomous learning agents—a task more closely related to robotics, and embodied cognitive sciences—has only recently been provided by some of the authors [11]. The latter work is embedded in the framework of projective simulation (PS) for AI, the central component of which is a specific memory system utilized by the agent. This memory system, called episodic and compositional memory (ECM) provides a platform for *simulating future action* before real action is taken. The ECM can be described as a stochastic network of so-called *clips*, which represent prior experiences of the learning agent, whose decision-making process is realized by a stochastic random walk in the clip space. In the agent’s design, it is the specific structure of the ECM

that is particularly suitable for quantization.

In this work we present a proposal for the experimental implementation of both classical and quantum PS agents in systems of trapped ions. While the classical variants of PS agents can easily be realized in physical systems without requiring quantum control, we show here how certain implementations of classical agents in ion traps, can—in a generic way—be used to construct quantum PS agents, through a nested process of *coherent controlization*.

The outline of this paper is as follows. In Section II we briefly review the PS model and give the basic operational elements which have to be constructed in an implementation of a classical or quantum PS agent. Then, in Section III we give a more formal treatment of the standard, classical PS agent, and show explicitly how such an agent may be implemented in an ion trap set-up. In particular, in Section III C, we discuss how the technique of coherent controlization provides a generic construction for emulating the standard PS agent in quantum systems, specifically in trapped ions. Finally, in Section IV, we extend our analysis to quantum PS agents by specifying all required operations and describing their implementation in ion traps. In the Appendix we further present a simple example for a quantum PS agent that can be straightforwardly implemented in an ion trap, for which we provide numerical simulations incorporating an appropriate error model.

II. PROJECTIVE SIMULATION

The central component of a PS agent, as introduced in Ref. [12], is the episodic and compositional memory (ECM), which can be formally represented as a stochastic network of *clips*. Clips represent the units of episodic

* vedran.dunjko@uibk.ac.at

† nicolai.friis@uibk.ac.at

‡ hans.briegel@uibk.ac.at

memory, which consist of memorized percepts, actions and ensuing rewards. The process of projective simulation is triggered by perceptual input that initiates a *random walk* over the clip space. This walk constitutes the stochastic replay of previously established memories and precedes the initiation of real action. The agent’s capability to learn is represented by two mechanisms, (i) the adaption of the transition probabilities between the clips, and (ii) the addition of new clips under compositional principles.

More formally, at any instance of time the ECM of an agent can be represented as a directed weighted graph, where the vertices represent the clips, and the weights of the edges represent the transition probabilities, see Fig. 1. We refer to this graph as the clip network. The random walk, or equivalently, the Markov chain, associated to the process of projective simulation is carried out over the clip network. Finally, the learning aspect of the agent is realized by updating the clip network based on the (re)actions and rewards of the environment, with which it interacts.

The criteria under which an action, that is, a clip representing a single memorized action in the ECM, is coupled out as real action can vary, leading to distinct types of PS agents. Here we list a few examples that we will encounter again later in this paper. In the so-called *standard* PS model, the first action clip that is encountered during the random walk over the clip network is coupled out as the chosen real action. The standard PS model can further be equipped with *emotion clips*, which are clip tags indicating, for instance, whether a chosen action recently lead to a reward. In this extended model, the random walk process can be iterated if the encountered action clip carries a ‘negative’ association—a process we will refer to as *reflection*.

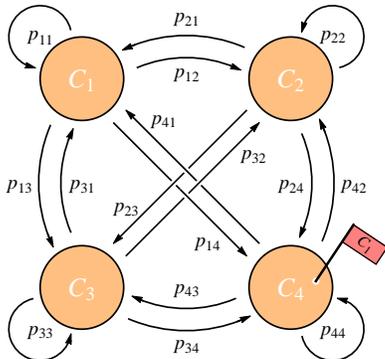


FIG. 1. **Clip network.** An example for a network with four clips c_i ($i = 1, 2, 3, 4$) is shown. At any fixed time, the PS agent associates a discrete-time homogeneous Markov chain with transition matrix $P = [p_{ij}]$ ($i, j = 1, 2, 3, 4$) to the ECM, which governs the transition probabilities for a random walk in the network. In addition, flags, here indicated on clip c_4 , may be introduced, e.g., to relate actions that were recently rewarded to the corresponding percepts.

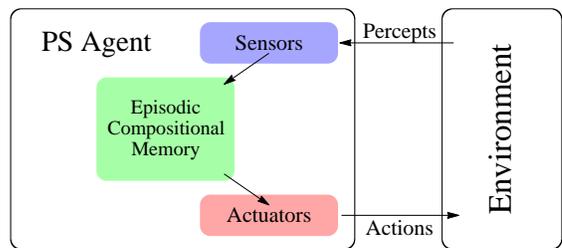


FIG. 2. **Projective simulation agent.** The (PS) model for active learning agents, introduced in Ref. [12], describes an embodied agent that interacts with its environment via sensory input (percepts), and action on the environment that is conducted using a set of actuators. The sensors and actuators are linked to the episodic compositional memory (ECM), which relates new perceptual input to the agent’s past experience.

Elaborating on the notion of reflection, in Ref. [11] some of the authors have recently introduced the *reflecting PS* (RPS) agent model, in which the Markov chain associated to the clip network is *ergodic*, and hence has a unique stationary distribution over the clip network. The RPS agent continues the random walk until the stationary distribution is reached, and (iteratively) samples from it until an action clip is observed. Building on the approaches of Refs. [13, 14] for quantizing random walks, this particular model was shown to have a quantum analog, called *quantum RPS*, which permits a quadratic speed-up in active learning scenarios.

As we have mentioned previously, the PS model can be endowed with additional structures, which further improve the agent’s learning capacity. These include the aforementioned emotion tags (a type of short-term memory), clip and edge glow (which allow for establishing temporal correlations), and other mechanisms, see, e.g., Ref. [15]. All of these additional structures are, in principle, compatible with the constructions we present, but for the most part we will not consider them here. We shall only utilize the simplest of these extensions, so-called flags, in the examples that are considered in the **Appendix**. As we will discuss, these flags allow for the demonstration of a quantum speed-up when incorporated into a very simple agent design, which is readily implementable in current laboratories.

In the next section, we present a more formal treatment of the standard PS model, and show how it can be implemented in an ionic set-up.

III. STANDARD PS AGENT

As noted, in the PS model, the ECM is represented as a clip network, that is, a weighted directed graph over the set of vertices $\mathcal{C} = \{c_i\}_{i=1}^N$, where each c_i represents a clip. The directed weighted edges of the graph represent the transition probabilities from one clip to an-

other given by a transition matrix $P = [p_{ij}]$ which is an $N \times N$ left-stochastic matrix, that is, $0 \leq p_{ij} \leq 1$ and $\sum_i p_{ij} = 1 \forall j$ ¹. In the standard PS, we can assume the clip network always contains clips which are representations of individual percepts (from the set of percepts $\mathcal{S} = \{s_i\}_i$) as well as clips that represent individual actions (from the set of actions $\mathcal{A} = \{a_j\}_j$), where $\mathcal{S} \cup \mathcal{A} \subseteq \mathcal{C}$ ². When presented with a percept s_i , the standard PS initiates a random walk in the clip network, governed by P , and starting from (the clip corresponding to) s_i . The walk is terminated at the first instance an action clip is encountered. This action is then coupled out as a real action.

This process can be viewed in terms of probability vectors as follows. Each clip c_i can be represented as a canonical basis vector of an N -dimensional real vector space \mathcal{V} , that is, $c_i = [0, \dots, 1, 0, \dots, 0]^T$, with the unity at the i^{th} position. The state after one random walk transition is

$$P c_i = \sum_j p_{ij} c_j, \quad (1)$$

which is a probability vector, i.e., a vector with real non-negative entries summing to one, representing a probability distribution over the clip space. This distribution is then sampled from, obtaining some clip c_k , which, if it represents an action, is coupled out. Otherwise the random walk proceeds from c_k .

In the spirit of the reinforcement learning paradigm, each round of interaction with the environment is either rewarded or not, and both cases lead to an update of the clip network, by altering the transition probabilities, and/or by altering the clip set itself, which constitutes the *learning aspect* of the PS agent. For an overview of the standard PS model, including examples of update rules, see Ref. [15].

A. Standard PS with Trapped Ions

We shall now discuss how the random walk initiated in an standard PS agent can be emulated in a quantum

system, in particular, using laser pulses on a string of trapped ions. Although a quantum implementation is not strictly required for the classical random walk of the standard PS agent, such a construction is the prerequisite for the fully quantized RPS agent that we will discuss in Section IV. For the construction of a quantum mechanical analogue of the transition matrix P we start by promoting the real vector space \mathcal{V} to a complex Hilbert space \mathcal{H} , and representing the clips c_i as orthonormal basis states $|c_i\rangle$. We then construct a unitary U_i , such that for a fixed basis state denoted $|0\rangle$ —this may correspond to some clip state $|c_l\rangle$ but the particular choice of this fixed state is unimportant—the components of the state $U_i |0\rangle$ with respect to the clip basis encode the transition amplitudes as dictated by the transition matrix P , i.e.,

$$U_i |0\rangle = \sum_{j=1}^N \sqrt{p_{ji}} |c_j\rangle. \quad (2)$$

We can see that a measurement of the state above in the clip basis recovers the right-hand side of the classical Eq. (1). However a single unitary cannot encode all the transitions of P . This can be seen quite simply, by noting that the columns of the matrix representation of U_i are required to be orthogonal, while the columns of P may even be identical. In general, one therefore requires N distinct unitaries U_i to represent all transitions of P on an N -dimensional Hilbert space. In other words, the first column, corresponding to the basis state $|0\rangle$, of the unitary U_i determines the transition probabilities from the clip c_i to any other clip in the sense of Eq. (2). Eq. (1) could be recovered even if the amplitudes in Eq. (2) had arbitrary relative complex phases. These phases are irrelevant in the context of the classical agent, but for the purpose of the extension to the quantum RPS we restrict the entries of the first column of U_i to be real and positive.

Note that, given the set of unitaries $\{U_i\}_{i=1}^N$, each corresponding to a column of an N -state transition matrix P , one can emulate any classical random walk by iterating the measurement of the quantum register (in the clip-basis), resetting the register to the state $|0\rangle$, and applying the U_i corresponding to the prior measurement result. The capacity to generate such unitaries will, in the next section, be used as a primitive to construct coherent quantum walks. Here we first analyze how such unitaries can be realized in an ionic set-up.

To proceed, we wish to encode the clip basis in the internal states of a chain of trapped ions, and the unitaries U_i in the laser pulses driving the transitions between them. We will consider a setup as described, e.g., in Ref. [16]. A string of k individual $^{40}\text{Ca}^+$ ions, with relative distances of some μm , is confined by a quadrupole trap (Paul trap). The ion confinement is best described by harmonic potentials, and the Coulomb repulsion of the ions couples the harmonic oscillators, such that the motion of the ions can be captured in

¹ Technically, since in the standard PS model, the action is coupled out whenever an action clip is hit, the probabilities of transiting from an action clip are undefined. However, we can, for simplicity, assign a unit probability of transiting to itself to each action clip. Thus, action clips are the absorbing states of the underlying Markov chain, although this will not be relevant for our work.

² In the last expression we have equated the representations of percepts and actions within the clip network with the actions and percepts themselves, in a slight abuse of notation. In the following, we will be using s_k (a_k) to denote the percept (action) clips when the semantics of the clip matters (e.g., whether it is an action or a percept), and the generic notation c_k when it does not. Formally, there is a distinction between percepts s_j and actions a_j , and their internal representation (a memory), usually denoted $\mu(s_j)$ and $\mu(a_j)$, respectively.

terms of their collective normal modes. For each ion, two Zeeman sub-levels, for instance, $|g\rangle := |S_{1/2, -1/2}\rangle$ and $|e\rangle := |D_{5/2, -1/2}\rangle$, which are related by a quadrupole transition, are used to represent the computational basis states $|0\rangle$ and $|1\rangle$ of a single qubit. In turn, we employ the state space of these k qubits as a representation of the clip network. Hence, the PS implementation we propose requires $k = \lceil \log_2(N) \rceil$ ions for a network of N clips.

The required unitaries can be realized with two laser beams [16], one of which is a broad beam that is nearly parallel to the linear chain, such that all ions are illuminated. Depending on the setting of the laser, two types of unitary gates can be implemented. When operated at the frequency ω corresponding to the transition $|0\rangle \leftrightarrow |1\rangle$ the laser realizes the collective single-qubit gate

$$U_X(\theta) = \exp(-i\frac{\theta}{2} \sum_{i=1}^k X_i), \quad (3)$$

where we use the shorthand notation X_i for $\mathbb{1}_1 \otimes \dots \otimes X_i \otimes \dots \otimes \mathbb{1}_k$, i.e., the Pauli X operator for the i -th qubit. On the other hand, when the laser is used in a bichromatic setting at two detuned frequencies $\omega_b > \omega$ and $\omega_r < \omega$, such that $\omega_b + \omega_r = 2\omega$, the Mølmer-Sørensen [17] gate

$$U_{\text{MS}}(\theta) = \exp(-i\frac{\theta}{4} \left[\sum_{i=1}^k X_i \right]^2) \quad (4)$$

is realized. The second, narrow laser beam, can be adjusted to address each ion individually, providing the single-qubit gates

$$U_{Z_i}(\theta) = \exp(-i\frac{\theta}{2} Z_i). \quad (5)$$

The operations of Eqs.(3)-(5) form a universal set of quantum gates, and hence provide the possibility to construct the unitaries U_i in principle. In general, the aim is to determine a sequence of operations with $(N-1)$ free parameters $\theta_1, \dots, \theta_{N-1}$, such that all entries of the first column of the resulting overall unitary $U(\theta_1, \dots, \theta_{N-1})$ are real and positive, and for appropriate choices of the θ_j their squares can form any arbitrary probability distribution $\{p_n\}_{n=1}^N$, with $\sum_n p_n = 1$. The freedom in the choice of parameters allows for all of the operators U_i to be represented by some specific choices of the θ_j . In particular, the agent is considered to operate based on a fixed internal architecture, in particular the tuning of the angles should have a simple operational meaning. At every step of the learning process, the agent only updates a set of parameters, here the θ_i , corresponding to the duration of some laser pulses within a fixed sequence. For instance, in the very simple case of a clip network with only two clips, the required unitary can be chosen to be a Pauli- Y rotation of a single qubit, given by

$$U_Y(\theta) = \exp(-i\frac{\theta}{2} Y) = \begin{pmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{pmatrix}, \quad (6)$$

which can be realized by three laser pulses, i.e.,

$$U_{Y_j}(\theta) = U_X(-\frac{\pi}{2}) U_{Z_j}(\theta) U_X(\frac{\pi}{2}), \quad (7)$$

and where we have included the qubit label j for later convenience.

As we have mentioned earlier, the ‘probability unitaries’ presented above will become the building blocks of the quantum PS agent. The second, and last, crucial ingredient in our construction is the technique of coherent controlization, which we shall briefly present next.

B. Coherent Controlization

A process of coherent controlization is any process by which we add quantum control [18] to individual elementary operations. More formally, we will call any process coherent controlization by which the implementations of some set of unitaries $\{U_i\}_{i=1}^M$, acting on a Hilbert space \mathcal{H} are combined to form one unitary U , acting on $\mathcal{H}_C \otimes \mathcal{H}$, where \mathcal{H}_C is an (at least) M -dimensional Hilbert space, such that

$$U|j\rangle \otimes |\psi\rangle = |j\rangle \otimes U_j|\psi\rangle, \forall |\psi\rangle \quad (8)$$

where $j = 1, \dots, M$, and $\{|j\rangle\}$ form an orthonormal basis of \mathcal{H}_C . This process forms an essential part of the construction of the quantum RPS agent that we will discuss in Section IV.

Moreover, coherent controlization provides an elegant method to generically assemble and combine probability unitaries. The latter may also be assembled in other, sometimes more efficient ways, and one alternative construction is provided in the Appendix. Nonetheless, the construction of the probability unitaries using coherent controlization offers the opportunity to illustrate this method on a simple and useful example.

Before we begin, let us recall the task at hand. For a given probability distribution $\{p_{ij}\}_{i=1}^N$, corresponding to the j -th column of the stochastic matrix P , we wish to construct the associated unitary U_j , such that the first column of U_j has real and positive entries $\sqrt{p_{ij}}$, with $i = 1, \dots, N$.

As the elementary operations that depend on these parameters we select single-qubit Y rotations $U_Y(\theta_i)$, which, for a trapped ion setup, may be realized as in Eq. (7), and where we drop the label Y for ease of notation. Any probability unitary $U(\theta_1, \dots, \theta_{N-1})$ on an N -clip network can then be assembled by a nested scheme of coherent controlization on k qubits, where k is the smallest integer that is larger than $\log_2(N)$. For simplicity, let us assume here that the size of the clip network is such that $\mathbb{N} \ni \log_2(N) = k$, which can always be achieved by duplicating some clips.

For a two-clip probability distribution $\{p_1, p_2\}$, the probability unitary is trivially realized by a single-qubit Y rotation $U(\theta_1)$, with $p_1 = \cos^2(\theta_1/2)$ and $p_2 = \sin^2(\theta_1/2)$. To extend this to a four-clip probability unitary $U(\theta_1, \theta_2, \theta_3)$, with probability distribution

$\{p'_1, p'_2, p'_3, p'_4\}$, one adds a second qubit, hence $k = 2$, and starts again with the operation $U(\theta_1)$ on the first qubit, where $p_1 = p'_1 + p'_2$ and $p_2 = p'_3 + p'_4$. This is followed by two controlled Y rotations of the second qubit, conditioned on the state of the first, that is, $U(\theta_2)$ is applied if the first qubit is in the state $|0\rangle$, while $U(\theta_3)$ is applied when the first qubit is in the state $|1\rangle$. The corresponding angles are determined from the renormalized probabilities within the respective subspaces, i.e., $\cos^2(\theta_2/2) = p'_1/(p'_1 + p'_2)$ and $\cos^2(\theta_3/2) = p'_3/(p'_3 + p'_4)$.

For larger values of k , the controlization becomes nested, see Fig. 3, e.g., for $k = 3$ ($N = 8$), the lowest level of single qubit operations, here $U(\theta_2)$ and $U(\theta_3)$, is followed by controlled operations on a third qubit. Labeling the qubits as I, II, and III, we may write the corresponding probability unitary as

$$U(\theta_1, \dots, \theta_7) = [U(\theta_2, \theta_4, \theta_5) \oplus U(\theta_3, \theta_6, \theta_7)]_{\text{I, II, III}} \times [U(\theta_1)_{\text{I}} \otimes \mathbf{1}_{\text{II, III}}], \quad (9)$$

where the controlled two-qubit operations are given by

$$U(\theta_2, \theta_4, \theta_5) = [U(\theta_4) \oplus U(\theta_5)]_{\text{II, III}} [U(\theta_2)_{\text{II}} \otimes \mathbf{1}_{\text{III}}], \quad (10a)$$

$$U(\theta_3, \theta_6, \theta_7) = [U(\theta_6) \oplus U(\theta_7)]_{\text{II, III}} [U(\theta_3)_{\text{II}} \otimes \mathbf{1}_{\text{III}}]. \quad (10b)$$

As we have argued above, coherent controlization allows for the construction of general probability unitaries from basic single-qubit probability unitaries. Despite the simple appearance of the circuits in Fig. 3, the practical implementation of coherent controlization requires additional attention. In fact, it is generally impossible to decompose quantum-controlled operations $\text{ctrl}(U)$ into individual gates $\text{ctrl}(U) = G_1 U G_2$, such that the G_i are independent of U , which implies that the gates G_i may not be specified if U is unknown [19, 20]. This seems

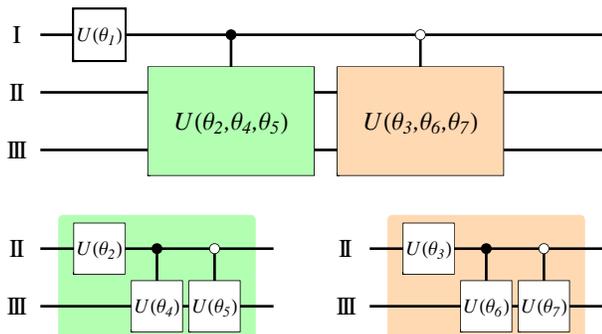


FIG. 3. **Coherent controlization.** The circuit diagrams show the construction of a three-qubit probability unitary (10), using coherent controlization. The filled dots “•” on the controlled operations indicate that the unitaries on the target are conditioned on the control qubit state $|0\rangle$, while the hollow dots “◦” represent conditioning on the control qubit state $|1\rangle$.

to suggest that coherent controlization requires computational effort in its implementation. However, for the ionic implementation that we will discuss next, we exploit additional degrees of freedom of the physical setup to perform coherent controlization in a generic way.

C. Coherent Controlization in Trapped Ions

We shall now discuss how quantum control can be practically added to unitaries that are realized by laser pulses in a trapped ion setup, based on the scheme introduced in Ref. [18]. As an example we give the explicit pulse decomposition that realizes the two-qubit unitary $U(\theta_1, \theta_2, \theta_3)$, which can be viewed as a special case of Eq. (10a) for $\theta_4, \dots, \theta_7 = 0$, where we use two ions, labeled I and II, respectively, before we explain how this method is generalized to the control of k -qubit unitaries.

To start, we note that the operation $[U(\theta_1)_{\text{I}} \otimes \mathbf{1}_{\text{II}}]$ can be trivially implemented by the pulse sequence of Eq. (7), and we can thus focus our attention on the remaining term $[U(\theta_2) \oplus U(\theta_3)]_{\text{I, II}}$. Apart from the laser pulses for the elementary operations $U(\theta_2)$ and $U(\theta_3)$, our scheme for their coherent controlization also consists of a number of additional Y rotations in 2-dimensional subspaces of the ionic energy levels other than the one spanned by $|g\rangle$ and $|e\rangle$, see Fig. 4. We will use additional superscripts, e.g., $U_{Y_i^\#}$, where the labels “#” identify different detuning frequencies, and the subscript $i \in \{\text{I, II}\}$ identifies the ion, to distinguish these operations. Furthermore, we make use of one of the common vibrational modes, which we assume has been cooled to the ground state $|0\rangle_v$, before the following steps are executed.

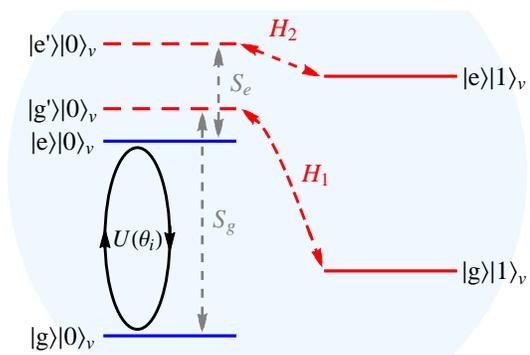


FIG. 4. **Level structure of trapped ions.** An illustration of the energy levels of one of the ions in the trap is shown. Two levels, $|g\rangle$ and $|e\rangle$, are chosen to represent the qubit, while the auxiliary levels $|g'\rangle$ and $|e'\rangle$, and the first excited state $|1\rangle_v$ of the common vibrational mode are used in the process of coherent controlization. The transitions indicated by H_1 , H_2 , S_g , and S_e can be realized by appropriately detuned Y -pulses.

- (i) Cirac-Zoller [21, 22] method: A sequence of appropriately blue-detuned laser pulses is applied on ion I to realize $U_{Y_I}^{CZ}(\pi)$, which transfers the population of $|g\rangle_I|0\rangle_v$ to $|e\rangle_I|1\rangle_v$. This step encodes the state of qubit I in the vibrational mode, i.e., the initial state of the form $(\alpha|g\rangle_I + \beta|e\rangle_I)|\psi\rangle_{II}|0\rangle_v$ is transformed to $|e\rangle_I|\psi\rangle_{II}(\beta|0\rangle_v + \alpha|1\rangle_v)$.
- (ii) Hiding: Red-detuned laser pulses corresponding to $U_{Y_{II}}^{H_1}(\pi)$ and $U_{Y_{II}}^{H_2}(\pi)$ are applied to ion II to transfer the populations from $|g\rangle_{II}|1\rangle_v$ to $|g'\rangle_{II}|0\rangle_v$, as well as from $|e\rangle_{II}|1\rangle_v$ to $|e'\rangle_{II}|0\rangle_v$, as illustrated in Fig. 4. Denoting the state ψ encoded in the levels $|g'\rangle_{II}$ and $|e'\rangle_{II}$ as $|\psi'\rangle_{II}$, we may write the overall state after this step as $|e\rangle_I(\alpha|\psi'\rangle_{II} + \beta|\psi\rangle_{II})|0\rangle_v$.
- (iii) $U(\theta_3)$: The pulse sequence that realizes $U(\theta_3)$ is applied to ion II, which leaves the system in the state $|e\rangle_I(\alpha|\psi'\rangle_{II} + \beta U(\theta_3)|\psi\rangle_{II})|0\rangle_v$.
- (iv) Switching: To exchange the primed and unprimed levels, laser pulses for $U_{Y_{II}}^{S_g}(\pi)$ and $U_{Y_{II}}^{S_e}(\pi)$, which are blue- and red-detuned, respectively, are applied to ion II, see Fig. 4. The resulting overall state after these operations is $|e\rangle_I(\alpha|\psi\rangle_{II} + \beta|(U(\theta_3)\psi')\rangle_{II})|0\rangle_v$.
- (v) $U(\theta_2)$: The pulse sequence that realizes $U(\theta_2)$ is applied to ion II, such that the system is now in the state $|e\rangle_I(\alpha U(\theta_2)|\psi\rangle_{II} + \beta|(U(\theta_3)\psi')\rangle_{II})|0\rangle_v$.
- (vi) Switching: The primed and unprimed levels are exchanged again using the laser pulses for $U_{Y_{II}}^{S_g}(\pi)$ and $U_{Y_{II}}^{S_e}(\pi)$ on ion II, which leaves the system in the state $|e\rangle_I(\alpha|(U(\theta_2)\psi')\rangle_{II} + \beta U(\theta_3)|\psi\rangle_{II})|0\rangle_v$.
- (vii) Unhiding: The hiding operations of step (ii) are reversed by the application of $U_{Y_{II}}^{H_1}(-\pi)$ and $U_{Y_{II}}^{H_2}(-\pi)$ to ion II, leaving the system in the state $|e\rangle_I(\alpha U(\theta_2)|\psi\rangle_{II}|1\rangle_v + \beta U(\theta_3)|\psi\rangle_{II}|0\rangle_v)$.
- (viii) Return control: Finally, $U_{Y_I}^{CZ}(-\pi)$ is applied to ion I, which returns the control from the vibrational mode, and provides the desired state $(\alpha|g\rangle_I U(\theta_2)|\psi\rangle_{II} + \beta|e\rangle_I U(\theta_3)|\psi\rangle_{II})|0\rangle_v$, that is, the unitary $U(\theta_2)$ acts on ion II, when ion I is in the state $|g\rangle_I$, while $U(\theta_3)$ acts upon the subspace in which the first ion is in the state $|e\rangle_I$.

If required, the scheme laid out in steps (i)-(viii) may be straightforwardly extended to larger clip spaces by increasing the number of control qubits and vibrational modes used. Each Y rotation in principle requires 3 individual pulses, see Eq. (7), but the collective X rotations for the operations $U(\theta_i)$ can be subsumed into two single pulses $U_X(\frac{\pi}{2})$ and $U_X(-\frac{\pi}{2})$ at the start and at the end of the entire pulse sequence, respectively. We hence find that the overall number of elementary laser pulses necessary to assemble a k -qubit probability unitary is given

by $(7 \times 2^{k+2} - 24k - 29)$ for $k \geq 2$. Note that an exponential scaling in terms of the qubits used is inevitable, as k qubits encode 2^k probabilities, and we must have the freedom to specify each one of these. In terms of the state space of the ECM network (clip number) the scaling is linear.

In such a process $(k-1)$ vibrational modes of different frequencies are used to generalize steps (i) and (viii) to condition $(k-1)$ -qubit operations on the state of the first qubit, i.e., by transferring the populations (exclusively) between $|g\rangle_I|0\dots 0\rangle_{v_1,\dots,v_{k-1}}$ and $|e\rangle_I|1\dots 1\rangle_{v_1,\dots,v_{k-1}}$.

Next, we give the basics of the classical and quantum RPS agent models, and show how the two components—coherent controlization and probability unitaries—can be utilized to construct these in systems of trapped ions.

IV. REFLECTING PS WITH TRAPPED IONS

We now turn to the so-called reflecting projective simulation (RPS) agent introduced in Ref. [11]. The central aim of the RPS is to output the actions according to a specific distribution, which we shall specify shortly, that is updated, indirectly, as the ECM network is modified throughout the learning process. Here, the clip network \mathcal{C} is disjoint, and it comprises unconnected percept-specific subnetworks with associated stochastic (ergodic and time-reversible) matrices $P_k = [(p_k)_{ij}]$, for each percept s_k .

Depending on which percept is observed, the random walk is executed on the corresponding percept-specific (sub-)network, where it is continued until the Markov chain P_k is (approximately) mixed, that is, until the respective stationary distribution π_{P_k} , which has support over the entire clip space, is (approximately) reached. The agent then samples from the obtained distribution, and iterates the procedure (which requires re-mixing of the Markov chain) until an action is hit. More specifically, the RPS agent is designed to output (a good approximation) of the *tailed distribution* $\tilde{\pi}_{P_k}$ defined as

$$(\tilde{\pi}_{P_k})_j = \begin{cases} \mathcal{N} \times (\pi_{P_k})_j, & \text{if } c_j \text{ is an action,} \\ 0, & \text{otherwise,} \end{cases} \quad (11)$$

where \mathcal{N} is a normalization factor such that $\sum_j (\tilde{\pi}_{P_k})_j = 1$. That is, the re-normalized distribution π_{P_k} truncated such that it has support only over the action space.

Despite the differences in the walk termination criteria of the standard PS and RPS models, all the operational elements required for an emulation of a classical RPS agent in an ionic set-up have already been presented in the last section, as the previously described construction enables the emulation of any classical random walk.

In the remainder of this section, we aim to show how the *quantum* RPS agent, which employs a truly coherent quantum walk (in the sense of [13, 14]) to obtain a

quadratic speed-up over the classical RPS agent, can be implemented based on the coherent controlization of unitaries as discussed in Section III C. For notational simplicity, we will from this point on ignore the subscript k indicating the percept the network in question corresponds to, unless it is specifically required.

The central process of the quantum RPS model, the basics of which we present next, is a so-called Szegedy-type quantum random walk, see, e.g., Ref. [14], that is performed on the percept-specific ECM (sub-)network. These Szegedy-type quantum random walks are used in the quantum RPS agent in order to output an action distributed according to the tailed stationary distribution $\tilde{\pi}_P$ with a quadratically decreased number of elementary diffusion steps, as compared to a classical RPS agent.

As the structure of this decision-making process is rather involved, let us briefly sketch it out here, before proceeding in more detail. The basic building block of a Szegedy-type walk, is the elementary *diffusion unitary* U_P , which acts on a two register system, each one of sufficient dimensionality to represent the entire clip network. One application of U_P can be considered as the analog of one step of the classical walk governed by the transition matrix P . The Szegedy walk operator $W(P)$, on the other hand, is constructed using four applications of U_P (or its inverse), and some quantum operations which are independent from P . One of the distinct properties of the operator $W(P)$ is that its unique (+1) eigenstate $|\pi'_P\rangle$ is a particular coherent encoding of the stationary distribution π_P of the Markov chain, as we clarify presently. Exploiting this property, and using a modified Kitaev phase estimation algorithm [23], we can construct an approximate reflection operator (ARO), which reflects over the state $|\pi'_P\rangle$. The speed-up achieved in the quantum RPS originates, in part, from the efficiency of the construction of the ARO operator in terms of the number of applications of the diffusion unitary U_P , relative to the mixing time of the Markov chain as specified by P .

The ARO operator above can then be used in search algorithms (e.g., as in Refs. [13, 14]), as well as in the decision-making process of the RPS agent, which can be seen as a Grover-type [4] reflection process in the following sense. Upon the system, initialized in the state $|\pi'_P\rangle$, one sequentially applies a ‘check’ operator, which adds a relative phase of (-1) to all basis states corresponding to actions, followed by the ARO operator, which reflects over the coherent encoding of the stationary distribution. This, like in the Grover algorithm, induces a sequence of rotations in a 2-dimensional workspace, which, after a certain number of iterations, guarantees that the system state has a constant overlap with the state encoding the aforementioned tailed distribution. The second component of the quantum speed-up lies in the number of these iterations, which inherits the quadratic improvement that is characteristic to Grover’s algorithm. With this in mind, let us now give further details of the building blocks of the quantum RPS.

A. The Szegedy Walk Operator

As we have argued previously, a unitary on an N -dimensional Hilbert space is generally not capable of representing all transitions of a Markov chain over a network of N clips. For this reason, the classical random walk for a given transition matrix P that we have described in Section III A is realized by, in general, N unitaries U_1, \dots, U_N , where U_i is associated with the i -th column of P . In the Szegedy-type approach to quantum walks, two copies, \mathcal{H}_I and \mathcal{H}_{II} , of an N -dimensional Hilbert space, i.e., $\dim(\mathcal{H}_I) = \dim(\mathcal{H}_{II}) = N$ are used to accommodate for all the required degrees of freedom. For a time-reversible Markov chain we define the unitary walk operators U_P and V_P as

$$U_P |c_i\rangle_I |0\rangle_{II} = |c_i\rangle_I U_i |0\rangle_{II}, \quad (12a)$$

$$V_P |0\rangle_I |c_i\rangle_{II} = (U_i |0\rangle_I) |c_i\rangle_{II}, \quad (12b)$$

where $\{|c_i\rangle_{I/II} | i = 1, \dots, N\}$ form bases of $\mathcal{H}_{I/II}$. The unitaries U_i act on $|0\rangle_{I/II} = |c_i\rangle_{I/II}$ according to

$$U_i |0\rangle_{I/II} = \sum_{j=1}^N \sqrt{p_{ji}} |c_j\rangle_{I/II}. \quad (13)$$

In the context of quantum RPS agents, we assume that the underlying ergodic Markov chain is time-reversible, i.e., it satisfies detailed balance. Although the Szegedy-type walk can be defined even if this is not the case, one would additionally require access to the time-reversed transition matrix³ P^* in such a situation. Here, we will present the construction in the most general terms, with the implicit understanding that for the RPS, the unitary V_P can be obtained from U_P by swapping the registers prior to, and after the application of U_P . With the operators U_P and V_P at hand, we can now proceed with the construction of the Szegedy walk operator $W(P)$, which is implemented by reflecting over the spaces A and B , defined as

$$A := \text{span}\{|\psi_i^A\rangle = U_P |c_i\rangle_I |0\rangle_{II} | i = 1, \dots, N\}, \quad (14a)$$

$$B := \text{span}\{|\psi_i^B\rangle = V_P |0\rangle_I |c_i\rangle_{II} | i = 1, \dots, N\}. \quad (14b)$$

The generalized walk operator is then defined as

$$W(P) = \text{ref}(B) \text{ref}(A), \quad (15)$$

where, for $X = A, B$, we have

$$\text{ref}(X) = 2 \sum_{i=1}^N |\psi_i^X\rangle \langle \psi_i^X| - \mathbf{1}_{N \times N}. \quad (16)$$

³ Note that the asterisk on the time-reversed transition matrix $P^* = (p_{ij}^*)$ does not indicate complex conjugation, and its components are given by $p_{ij}^* = p_{ji}(\pi_P)_i / (\pi_P)_j$.

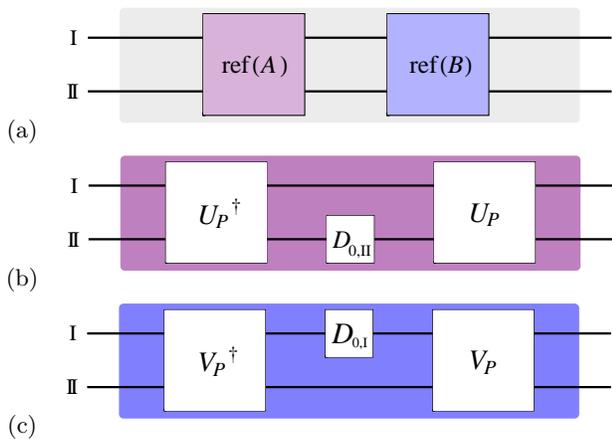


FIG. 5. **Szegedy walk operator.** The circuit representations of the Szegedy walk operator $W(P)$ of Eq. (15), as well as the reflections over A and B [see Eq. (14)] are shown in Fig. 5 (a), (b), and (c), respectively. The reflection over A (B) is fully determined by the walk operator U_P (V_P) and a reflection over $|0\rangle$, i.e., $D_0 = 2|0\rangle\langle 0| - \mathbb{1}_N$.

The two operators $\text{ref}(A)$ and $\text{ref}(B)$ are constructed from the diffusion operators, U_P and V_P , along with reflections over $|0\rangle_I$, and $|0\rangle_{II}$ denoted $D_{0,I}$ and $D_{0,II}$, respectively, as shown in Fig. 5. The unique (+1) eigenstate $|\pi'_P\rangle$ of the Szegedy walk operator $W(P)$, which coherently encodes the stationary distribution π_P on the two registers, is given by

$$|\pi'_P\rangle = U_P |\pi_P\rangle_I |0\rangle_{II} = \sum_i \sqrt{(\pi_P)_i} |c_i\rangle_I U_i |0\rangle_{II}. \quad (17)$$

B. The Approximate Reflection Operator

The next step in the design of a quantum RPS agent is the construction of the *approximate reflection operator* (ARO) from the walk operator $W(P)$. The ARO operator is designed to approximate the (ideal) reflection operator

$$\text{ref}(|\pi'_P\rangle) = 2 |\pi'_P\rangle\langle\pi'_P| - \mathbb{1}_{I,II}. \quad (18)$$

With the generalized walk operator $W(P)$ at hand, an approximate reflection over $|\pi_P\rangle$ is obtained [14] by implementing $PD(W)$, a modification of Kitaev's [23] phase estimation algorithm, shown in Fig. 6. For this task, we add $(n+1)$ ancilla qubits, where n scales as $\log_2(1/\sqrt{\delta})$, where $\delta = 1 - |\lambda_2|$ is the spectral gap of the Markov chain, i.e., λ_2 is the second largest eigenvalue of P . We employ $PD(W)$ and its inverse operation, with an intermediate reflection over the ancilla state $|00\dots 0\rangle_{\text{Aux}}$. This combination of operations approximates the reflection over $|\pi'_P\rangle$ from Eq. (18). An analysis of the fidelity of the reflection, as a function of n , is given in Ref. [14]. The crucial feature of this construction is that the ARO operates based on a number

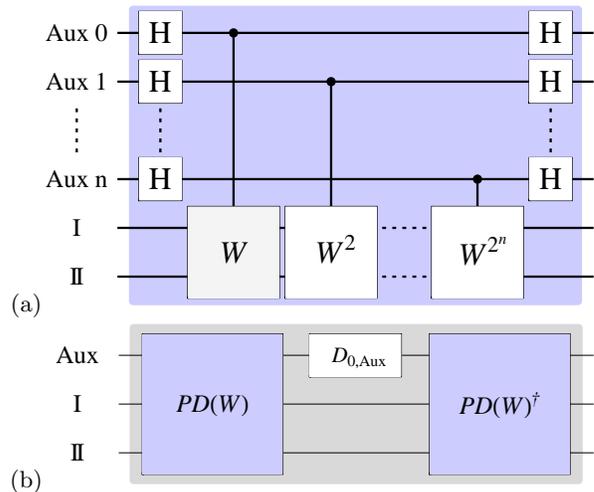


FIG. 6. **Phase detection and approximate reflection.** The circuit in Fig. 6 (a) shows the phase detection operator $PD(W)$, which forms part of Kitaev's phase estimation scheme [23]. Registers I and II are complemented by $(n+1)$ ancilla qubits, here labeled $\text{Aux } 0, \text{Aux } 1, \dots, \text{Aux } n$, which are all initialized in the state $|0\rangle_{\text{Aux } i}$ ($i = 0, \dots, n$), followed by Hadamard gates $H_{\text{Aux } i}$. The executions of the (2^m) -th power of W_k is then conditioned on the state of qubit $\text{Aux } m$, before another Hadamard gate is performed. In Fig. 6 (b) the approximate reflection operator (ARO) is combined with the phase detection circuit $PD(W)$ and its inverse $PD(W)^\dagger$, with an intermediate reflection over the ancilla state $|00\dots 0\rangle_{\text{Aux}}$.

of calls to $W(P)$ that scales as $\tilde{O}(1/\sqrt{\delta})$ ⁴, while the number of calls to P to prepare the stationary distribution for the classical RPS scales as $\tilde{O}(1/\delta)$.

C. Quantum deliberation

To output a distribution of actions that corresponds to the tail of the stationary distribution with support only over the (flagged) actions, the agent performs a quantum deliberation process with elements reminiscent of Grover-like steps [4, 14]. In the preparation phase, the agent first initializes the joint system of registers I and II in the state $|\pi'_P\rangle$ from Eq. (17). While the preparation of this initial state may be involved in general, in certain cases, including the one presented in the appendix, it becomes straightforward. Consecutively, the agent alternately applies the following two operations:

- (i) Reflection over the actions:

$$\text{ref}(\mathcal{A}) = 2 \sum_{i \in \mathcal{A}} |c_i\rangle\langle c_i|_I - \mathbb{1}_I, \quad (19)$$

where \mathcal{A} denotes the set of (flagged) actions.

⁴ The tilde-O notation designates that, for this analysis, we are ignoring factors which are contributing only logarithmically.

(ii) Approximate reflection over the state $|\pi'_p\rangle$.

The sequence of operations above will, similarly to Grover's algorithm, increase the amplitude of the actions with respect to non-action components in the state of the system, while maintaining the relative weights of the action elements. This ensures that the actions are output according to the correct distribution, as explained in [11].

After iterating these steps a number of times that is determined by the relative probability $\epsilon = \sum_{i \in \mathcal{A}} (\pi_p)_i$ of the actions within the stationary distribution, the agent samples, that is, measures in the clip basis of register I. If a desired action is found, it is coupled out, otherwise the procedure is repeated [11]. The average number of iterations of the Grover-like steps (i) and (ii) scales as $\tilde{O}(1/\sqrt{\epsilon})$, while the classical RPS agent requires $\tilde{O}(1/\epsilon)$ iterations on average.

D. Reflecting PS Implementation for Trapped Ions

Finally, let us examine the possibility to implement the decision-making process of a quantum RPS agent in an ion trap. As we have explained, two operators are required, the reflection over (flagged) actions, and the ARO. The former can be generically achieved, for instance, by applying the detuned pulses corresponding to $U_{Y_i}^{S_g}(2\pi)$ or $U_{Y_i}^{S_g}(\pi)$ of the coherent controlization step (iv) specifically to those basis states corresponding to (flagged) actions, flipping their sign. The latter, the ARO, is implemented starting from the probability unitaries, by coherent controlization, in conjunction with a few fixed operations, $D_{0,I}$, $D_{0,II}$, $D_{0,Aux}$ and H .

Let us briefly describe the individual steps of this procedure. By coherently conditioning the probability unitaries U_i , the operation U_P is obtained, from which the pulse sequence for V_P is obtained by swapping the registers, which, in practice, corresponds to an exchange of the qubit/ion labels in the pulse sequence for U_P . The associated inverse operators follow immediately by setting $\theta_i \rightarrow (-\theta_i)$. The reflections $D_{0,I}$, $D_{0,II}$, and $D_{0,Aux}$ are obtained as special cases of the reflection over the (flagged) actions. The Hadamard gate

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad (20)$$

can be implemented up to a phase of $(-i)$, that is, for the j -th ion we have the pulse sequence

$$-iH = U_X(-\frac{\pi}{2})U_{Z_j}(\frac{\pi}{2})U_X(\frac{\pi}{2})U_{Z_j}(\pi), \quad (21)$$

with U_X as in Eq. (3), and U_{Z_j} given by Eq. (5). The superfluous phase $(-i)$ cancels naturally, since the Hadamard gate is used four times for every ancilla in the ARO, twice each for the realization of $PD(W)$ and its inverse, see Fig. 6. Finally, we make again use of coherent controlization to construct the phase detection operator $PD(W)$ and its inverse from the walk operator $W(P)$. The possibility to add control to arbitrary

(unknown) unitaries hence provides a modular structure, that allows, in principle, for the generic implementation of all operations that required for the decision-making of a quantum RPS agent. The modular use of coherent controlization in the design of the agent can thus be summarized by the following sequence:

$$U_Y(\theta_i) \xrightarrow{CC} U_j(\theta_1, \dots, \theta_{N-1}) \xrightarrow{CC} U_P, W(P) \xrightarrow{CC} \text{ARO}.$$

That is, starting from single qubit Y rotations, parameterized according to the stochastic matrix P , we construct the probability unitaries using coherent controlization. From the probability unitaries we then construct, again by coherent controlization, U_P and V_P , which are used to assemble $W(P)$. Finally, from $W(P)$ we construct the ARO operator that is central to the quantum deliberation steps, once again employing coherent controlization.

Despite the fact that all individual operations of the quantum RPS are implementable with current technology, we note that very large network sizes, as well as small values of ϵ or δ would push the limit of the state-of-the-art for ionic implementations. Nonetheless, special cases of the general scheme we have laid out here are well within reach of experimental testing. In the [Appendix](#), we present such an example for a quantum RPS agent based on an ECM using two qubits, and we give an explicit pulse decomposition of its entire decision-making process, including an error analysis.

V. CONCLUSIONS

We have presented a modular architecture for the implementation of the deliberation process of PS agents in systems of trapped ions. We have shown first how the probability unitaries, which are required for the emulation of classical random walks, can be generically constructed using coherent controlization, and second how this process allows for the implementation of a quantum RPS agent based on these probability unitaries. A main feature of our construction is its modular architecture, that is, any changes of the probabilities as part of the learning process can be dealt with at the level of the implementation of the probability unitaries, whereas the rest of the construction is unaltered. The generic construction relies only on elementary single-qubit Y rotations and coherent controlization, which allows for a straightforward assembly, as well as straightforward updating of the probability unitaries.

This is an important advantage, if not a prerequisite, for the realization of a learning agent that is continuously adjusting the probabilities underlying its deliberation process. Having to re-compute the entire sequence of gates which need to be applied to realize the quantum RPS agent for any change of the underlying Markov chain would impose a large computational overhead on the agent, and significantly diminish the advantage in speed that is provided by quantizing the RPS agent.

In addition to the general modular architecture, we have provided numerical simulations of an implementation of simple RPS agents using trapped ions. As our investigation shows, proof-of-principle realizations of these agents are simple enough to be implementable in current experimental setups, while they are sufficiently involved to demonstrate the quadratic speed-up.

ACKNOWLEDGMENTS

We are grateful to Adi Makmal, Markus Tiersch, Benjamin P. Lanyon and Daniel Nigg for valuable discussions and comments. HJB acknowledges discussions with Gavin Brennen at an early stage of this project. This work has been supported in part by the Austrian Science Fund (FWF) through the SFB FoQuS: F4012 and the Templeton World Charity fund grant TWCF0078/AB46.

Appendix: Rank-One Reflecting PS in Ion Traps

Here, we provide an example for a quantum RPS agent sophisticated enough for the demonstration of a quantum speed-up, whilst being sufficiently simple to allow an immediate implementation in readily available ion trap setups, e.g., as described in Ref. [16]. The Appendix is structured as follows. In Section 1 we first discuss the simplified decision-making process for a quantum RPS agent whose underlying ECM network corresponds to a rank-one Markov chain. To provide context, the role of these simple agents is then illustrated for the invasion game in Section 2. In Section 3, we propose an ion trap implementation of the rank-one quantum RPS agent, for which we supply the explicit overall pulse sequence. We accompany our proposal with an appropriate error model, and corresponding numerical simulations, which are given in the final Section 4.

1. Rank-One Reflecting PS

A special case of the RPS agents that we have considered in Section IV is obtained by considering the reflective analog of so-called “two-layered” PS agents, where all transition are one-step transitions from percepts to actions [11]. Such agents have a very simple structure, yet were shown to be capable of learning to solve non-trivial environmental tasks [15, 24]. In the RPS analog of two-layered PS agents [11], the associated Markov chains of each percept-specific clip network are rank-one throughout the entire learning process of the agent. The columns of P are then all identical, and equal to the stationary distribution. The spectral gap is given by $\delta = 1$, and the Markov chain mixes in one step. Let us consider the consequences—radical simplifications—for the construction of the RPS agent.

In the rank-one case, the probability unitaries U_i for a fixed P are all the same, so we can remove the subscript,

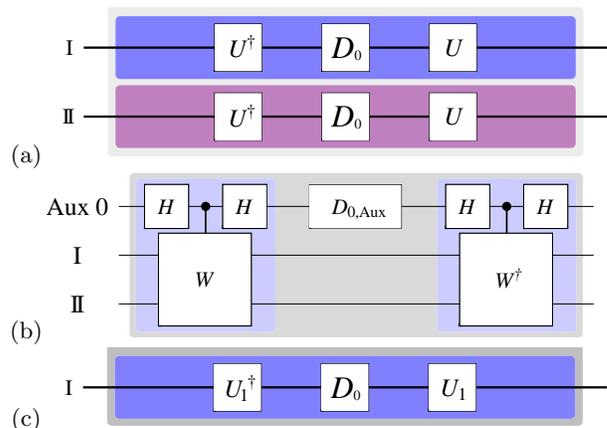


FIG. 7. **Rank-one reflection operator.** For rank-one Markov chains, U_P and V_P are local operations on registers II and I, respectively. The Szegedy walk operator $W(P)$ that is shown in Fig. 7 (a) hence factorizes into two independent applications of $U D_0 U^\dagger$. Since the walk operator further becomes Hermitean, $W = W^\dagger$, the single remaining ancilla is also redundant, the approximate reflection circuit shown in Fig. 7 (b) reduces to one application of $W(P)$ as shown in Fig. 7 (c), and the reflection becomes exact.

write only U , but we keep in mind the distinction of U and U_P . Moreover, coherent controlization is no longer necessary for the construction of U_P , since U is applied regardless of the state of the control register, $U_P = \mathbb{1} \otimes U$ ($V_P = U \otimes \mathbb{1}$). As can be easily seen, the reflections $\text{ref}(A)$ and $\text{ref}(B)$ shown in Fig. 5 then commute, acting locally on registers II and I, respectively, see Fig. 7. Similarly, the coherent encoding of the stationary distribution is now given by the product state $|\pi'_P\rangle_{I,II} = |\pi_P\rangle_I |\pi_P\rangle_{II}$.

When assembling the phase detection operator $PD(W)$ and the approximate reflection operator (ARO), see Fig. 6, the spectral gap of $\delta = 1$ means that (at most) one ancilla qubit is required. Now, note that the walk operator $W(P)$ for rank-one matrices P , as shown in Fig. 7 (a), is Hermitean, and thus the entire circuit shown in Fig. 7 (b) reduces to a single application of the Szegedy walk operator $W(P)$. An *exact* reflection over $|\pi_P\rangle$ can hence be performed by applying $W(P) = U D_0 U^\dagger$ to either of the registers, see Fig. 7 (c). Without loss of generality we select register I, where we drop the subscript indicating the register from now on, to perform all the Grover-like steps to output actions according to the tailed stationary distribution, which entails the following steps.

In the preparation stage, the state $|\pi_P\rangle$ is initialized by one application of U to the state $|0\rangle$. Then, the two operators of the Grover-like process, i.e., the reflection over the action $\text{ref}(A)$, and the reflection over $|\pi_P\rangle$, are applied a prescribed number of times determined by ϵ , the relative probability of the actions within the stationary distribution. Consecutively, the agent measures in the clip basis. If the measurement provides an action, it is coupled out, otherwise the agent iterates this procedure.

Before we continue with the ionic implementation of the deliberation process, let us briefly examine an example for a task—the invasion game—for which the agent may employ its capabilities of learning and decision-making.

2. The Invasion Game

As a simple example that can be solved by two-layered agents, let us discuss the invasion game, as considered in Ref. [12]. In this game, the agent is tasked with guarding a region of space from an adversary who attempts to enter the region through an array of entrances, see Fig. 8. The agent’s goal is to prevent the adversary from entering by blocking sites. In every round of the game, the adversary has three possible moves. It may attempt to enter at its current location, or move one door to the left, or one door to the right and attempt to enter through one of these openings. The agent is rewarded if it matches the move, thus blocking the adversary.

To emphasize the learning aspect of the game, we assume that the game starts with the adversary and the agent located at the same entrance, and before the adversary moves, it displays some signal that indicates which way he intends to move next. Thus, the set of percepts of the agent (the defender) is $\{\downarrow, \leftarrow, \rightarrow\}$, which hint at the possible subsequent move of the attacker. The agent itself can also choose to remain where it is, move left, or move right in an attempt to block, corresponding to the three action clips $c_1(a_\downarrow)$, $c_2(a_\leftarrow)$, and $c_3(a_\rightarrow)$ accessible to the agent.

For the RPS agents discussed previously, this simple game may be represented by associating a three clip network to each of the percepts. In what follows, we shall only focus on a network associated to one percept, say “ \downarrow ”, as everything will also hold for other subnetworks as well, and we shall drop the corresponding subscript for ease of notation. For such two-layered settings there is a simple construction relating the probabilities of outputting a particular action, and the structure of the underlying percept-specific Markov chain. In particular, the action probabilities $\pi = (\pi_1, \pi_2, \pi_3)$ are realized by the stochastic matrix where each column is the vector π . The learning of the agent manifests in the relative increases of probabilities corresponding to rewarded actions, and examples for specific update rules can be found, e.g., in Ref. [12].

In basic two-layered settings in both the RPS and the analogous standard PS agent models, an action is coupled out after exactly one diffusion step. In order to illustrate a speed-up in such a scenario, we therefore need to consider some additional structure that increases the learning efficiency of the agent, but induces a longer deliberation time. Such a structure can be provided by percept-specific *flags*, which correspond to rudimentary emotion tags. Flags can be interpreted as the agent’s short term memory, indicating favored actions. In other

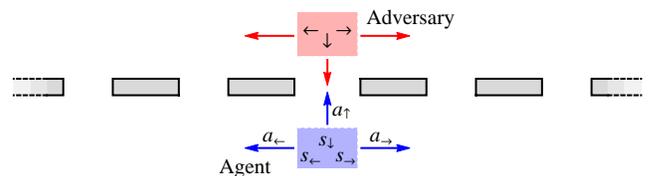


FIG. 8. **Invasion game.** In the invasion game [12] the agent defends a region of space against an adversary that tries to enter through a series of openings. To be rewarded, the agent is to prevent the adversary from entering, by blocking the passages, which can be achieved if the adversary’s signals, “ \downarrow ”, “ \leftarrow ”, and “ \rightarrow ”, indicating its next move, are interpreted correctly, and the agent mirrors the adversary’s moves.

words, absent flags indicate that a particular choice of action, for a given percept, was not rewarded in the previous step, and should be avoided. More precisely, this structure works as follows. Initially, all the actions are flagged. Then after an action has been coupled out, the flag is removed if the action is not rewarded. If the unflagged action is selected again after encountering the same percept in a consecutive round, the deliberation process is repeated until the deliberation results in a flagged action. In the case that the last remaining flag is removed, which indicates a definite change in the setting of the environment, all flags are re-set.

This structure leads to great improvements in settings where the environment (e.g., the adversary in the invasion game) changes its strategy, for instance, by permuting the meaning of the percepts [12]. In this case, if the network is already well-taught, the probability of outputting the correct action, once the meaning of percepts has been altered, can be very low. We will be interested in precisely such a setting. Suppose the attacker pursues a consistent strategy for a prolonged period of time, and the agent has learned well. This entails that, for a given percept, one of the values in the distribution $\pi = (\pi_1, \pi_2, \pi_3)$, say the third, is much larger than the others, e.g., $\pi_3 \ll \epsilon = \pi_1 + \pi_2$, and only the action clip corresponding to π_3 is flagged. Now, if the environment is to suddenly change its strategy, no longer rewarding this action, the flag on this clip will disappear, while flags on other clips are introduced again. Subsequently, the agent is required to output the tail of the distribution π with support only over the actions corresponding to π_1 and π_2 . However, for the classical RPS model, as well as for the standard PS model, the average number of iterated diffusion steps required until one of the remaining flagged actions is hit is $O(1/\epsilon)$, which can be exceptionally large, if the network was well-taught. The quantum variant of the RPS will then be quadratically faster, only requiring $O(1/\sqrt{\epsilon})$ steps. In any given round, the decision-making process after encountering a percept can then be represented on a two-qubit Hilbert space according to Table I.

Next, we discuss how a rank-one quantum RPS deliberation process based on this two-qubit system can be represented using two trapped ions.

clip	interpretation	two-qubit state
c_1	action a_\downarrow	$ 00\rangle$
c_2	action a_\leftarrow	$ 01\rangle$
c_3	action a_\rightarrow	$ 10\rangle, 11\rangle$

TABLE I. Representation of three-clip network as two qubits. A two qubit system can represent four clips, but as the desired network only requires three, a redundancy is introduced, e.g. in clip c_3 .

3. Rank-One Quantum RPS with Trapped Ions

To implement a rank-one quantum RPS agent for a setting such as the one described above, we construct the two-qubit operations U , D_0 and $\text{ref}(\mathcal{A})$, where the latter operation is now a reflection over flagged actions only, from laser pulses on two trapped ions. As we have described in Section III B, coherent controlization may be employed to assemble the probability unitary U , but in this simple case we may resort to a simpler option. As shown in Table I, we operate on a two-qubit Hilbert space, but we only distinguish between three clips, such that only two independent angles, θ_1 and θ_2 , parameterize the probability unitary U . A pulse sequence that achieves this is given by

$$U(\theta_1, \theta_2) = U_X(-\frac{\pi}{2}) U_{Z_2}(2\theta_2) U_{Z_1}(2\theta_1) U_X(\frac{\pi}{2}), \quad (\text{A.1})$$

where the collective X and single-qubit Z pulses are realized by individual laser pulses as described in Section III A. In terms of the probabilities π_1 and π_2 , which we assume correspond to the two flagged actions, the angles θ_1 and θ_2 are given by

$$\theta_1 = \arccos \sqrt{\pi_1 + \pi_2}, \quad (\text{A.2a})$$

$$\theta_2 = \begin{cases} \arccos \sqrt{\frac{\pi_1}{\pi_1 + \pi_2}}, & \text{for } \pi_1 + \pi_2 \neq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (\text{A.2b})$$

For the implementation of $\text{ref}(\mathcal{A})$, the reflection over the actions, one simply applies the single-qubit Z operation

$$U_{Z_1}(\pi) = \text{diag}\{-i, -i, i, i\}. \quad (\text{A.3})$$

Since the rank-one RPS operates solely on one register, the overall phase of the reflection is irrelevant, as long as the relative sign between flagged actions and all other clips is flipped. Finally, we propose the following implementation of D_0 . A detuned Mølmer-Sørensen pulse, see Eq. (4), is used to transfer the population of the state $|gg\rangle$, corresponding to $|00\rangle$, to an auxiliary state $|g'g'\rangle$. While the state $|00\rangle$ is hidden in this way, a single-qubit Z pulse $U_{Z_1}(2\pi)$ flips the sign of all other basis states, before a second Mølmer-Sørensen pulse returns the population to $|00\rangle$.

Taken together, all operations for one iteration of the Grover-like reflection may hence be realized by 12 laser pulses. In addition, 4 individual pulses are needed for the preparation of the initial state. At last, in the next section, we investigate the performance of our ion-trap quantum RPS agent in a series of numerical simulations that incorporate a suitable error model.

4. Numerical simulations

For the numerical simulations that we present in this final section, let us consider the most worrying sources of errors that may occur in an ion trap. On one hand, we consider imprecisions in the laser pulse frequency or duration, resulting in varying angles for the laser pulses we consider. We model such errors by randomly varying the angles for each pulse in the sequence according to a Gaussian distribution with standard deviation σ that is centered around the correct value. Addressing errors in principle represent another source of imprecision, but we assume that these may be largely corrected for by small modifications of the durations of subsequent laser pulses.

In the simulations, we specify a pair of values $\pi_1 > 0$ and $\pi_2 > 0$, such that $\epsilon = \pi_1 + \pi_2 < 1$, initialize the corresponding state vector $|\pi_P\rangle = U(\theta_1, \theta_2) |0\rangle$, and apply the combination of the reflections $\text{ref}(\mathcal{A})$ and $U D_0 U^\dagger$ a total of m times, where $m \in \mathbb{N}$ is chosen randomly from the interval $[0, m_\epsilon]$, with $m_\epsilon = \lceil 1/\sqrt{\epsilon} \rceil$. The clips are then randomly sampled according to the probability distribution

$$\left\{ |\langle c_i | [U D_0 U^\dagger \text{ref}(\mathcal{A})]^m U |0\rangle|^2 \right\}_i, \quad (\text{A.4})$$

which corresponds to a measurement in the clip basis. If no flagged action is found, a new number m is generated, and the procedure is iterated until a flagged action has been sampled. For every fixed set of π_1 and π_2 the process is repeated for 10^4 runs to build up statistics, out of which N_1 (N_2) result in an output of the action clip c_1 (c_2), corresponding to π_1 (π_2). Additionally, the overall number N_U of calls to the operator U until a flagged action is observed is recorded in each run.

For N_U the expected scaling as $(1/\sqrt{\epsilon})$ is largely independent from the error rates, as can be seen from Fig. 9, since this behavior is governed by the structure of the process, in particular, the upper bound m_ϵ for the randomly chosen value m . The integer steps by which m_ϵ increases, as $(1/\sqrt{\epsilon})$ decreases, also explain the step-like pattern visible in the data of Fig. 9. That is, in such a Grover-like scheme, the probability to sample a flagged action grows monotonically with the number of iterations only up to some point, from which on additional applications of the reflections will alternately decrease and increase the probability. The average number of repetitions set by the value m_ϵ , which corresponds to a fixed interval of ϵ -values, is hence not optimal for all ϵ within that interval, which can be seen

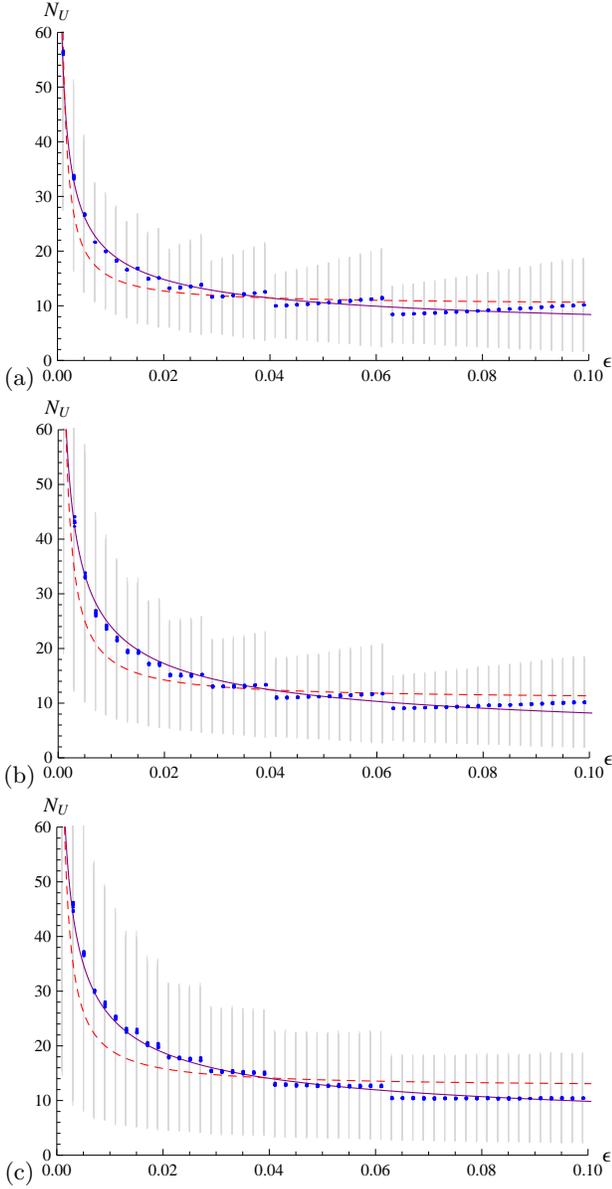


FIG. 9. **Average number of calls to U .** The results of the numerical simulation for the average number of calls to the probability unitary U until an action clip is hit are shown for error rates $\sigma = \pi/100$, $\sigma = \pi/20$, and $\sigma = \pi/10$, in Figs. 9 (a), (b), and (c), respectively. Each blue dot corresponds to the average over 10^4 runs for a fixed value $\epsilon = \pi_1 + \pi_2$, and the vertical gray lines indicate one standard deviation in each direction. The solid purple curves show the best fits that are linear in $(1/\sqrt{\epsilon})$, while the dashed red curves show the best fits that are linear in $(1/\epsilon)$, and we have confirmed that the former fit the data better than the latter.

from the slanting of the data points, and their standard deviations, in each of the ‘steps’ seen in Fig. 9 (a). The errors partially cover this effect, as can be seen in Figs. 9 (b) and (c).

To illustrate the speed-up of the quantum RPS agent with respect to a classical RPS agent, we directly com-

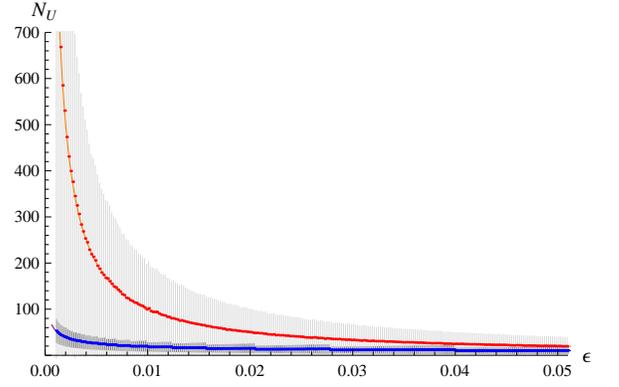


FIG. 10. **Comparison of classical and quantum RPS.** Numerical simulations of classical (upper red data points) and quantum RPS (lower blue data points) agents are shown. The data points are obtained as averages over 10^4 runs for each value of ϵ , and the corresponding standard deviations are indicated by the (light-)gray vertical bars. The fitted curves are linear in $(1/\epsilon)$ (top orange curve) and $(1/\sqrt{\epsilon})$ (bottom purple curve), respectively.

pare their performance in a simulation without errors, that is, for $\sigma = 0$, see Fig. 10. The classical rank-one RPS agent is emulated here by running the rank-one quantum RPS deliberation process described in this section for $m_\epsilon = 0$, that is, the state $U|0\rangle$ is prepared, and a sample is taken, such that clip c_i is obtained with probability $|\langle c_i U|0\rangle|^2$. If no flagged action is obtained, the procedure is repeated.

What remains to be confirmed by the simulations is the output of flagged actions according to the tail of the stationary distribution, as predicted in Ref. [11]. We address this question in two ways. First, we evaluate the behavior of a few selected illustrative pairs of probabilities π_1 and π_2 for increasing error rates in Fig. 11. As a measure for the accuracy of the output, we use the statistical distance

$$D(\tilde{\pi}, \tilde{N}) = \frac{1}{2} \sum_{i=1,2} \left| \frac{\pi_i}{\pi_1 + \pi_2} - \frac{N_i}{N_1 + N_2} \right|, \quad (\text{A.5})$$

of the output distribution $\tilde{N} = \{N_i/(N_1 + N_2)\}_{i=1,2}$ and the tailed stationary distribution $\tilde{\pi} = \{\pi_i/(\pi_1 + \pi_2)\}_{i=1,2}$. In Fig. 12 we then compare the relative frequencies N_1/N_2 with which the two flagged actions were obtained to the corresponding ratios π_1/π_2 of the (tailed) stationary distribution, for a broad range of values π_1 and π_2 , and for the three error rates previously chosen used in Fig. 9.

The data shown in Fig. 11 illustrates that large errors result in an output according to a uniform distribution over the flagged actions. The farther the tailed stationary distribution is away from the uniform distribution, the smaller the tolerance for errors. As the stationary distribution is updated throughout the learning process the errors will thus cause a stronger deviation from the desired output distribution.

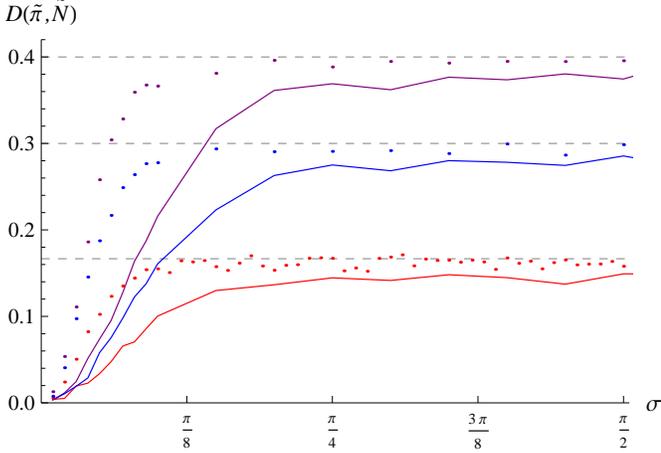


FIG. 11. **Statistical distance to tailed distribution.** The statistical distance $D(\tilde{\pi}, \tilde{N})$, see Eq. (A.5), of the output from the tailed stationary distribution is plotted against the width σ of the error distribution, for values $\epsilon = 0.05$ (solid) and 0.001 (dots), and ratios $\pi_1/\pi_2 = 9, 4$, and 2 (top to bottom). The dashed horizontal lines indicate the statistical distance to the uniform distribution for each pair $\{\pi_1, \pi_2\}$, which is approached when the errors dominate the behavior of the agent.

To make these statements more meaningful in terms of learning agents, let us consider a specific example. Let us assume that for a fixed percept, the tailed stationary distribution may be biased towards the action clip c_1 , such that an ideal agent outputs this action in 90% of the cases⁵. To reach this goal, such an agent updates the corresponding Markov chain throughout the learning process, until the associated stationary distribution is such that $\pi_1/\pi_2 = 9$. We may then set an error threshold, by assuming that the agent is still considered to succeed, if the action c_1 is performed only 70% of the time, i.e., a statistical distance of 20%. Brief inspection of the topmost solid curve in Fig. 11 reveals that for $\epsilon = 0.05$ the threshold value corresponds roughly to the largest error, $\sigma = \pi/10$, that we consider in Fig. 9. This, in turn, suggests a maximal number of $m_\epsilon = 5$ coherent iterations of the reflections in the Grover-like process before a measurement is performed, which translates to 64 individual laser pulses as described in Section 3.

The initial analysis presented in this appendix suggests that our proposal for the implementation of two-layered quantum RPS agents may be feasible, and be readily im-

⁵ The so-called ‘forgetfulness parameter’ γ which controls at what rate the agent is, roughly speaking, forgetting what it has learned, which radically speeds up re-learning [15] also implies

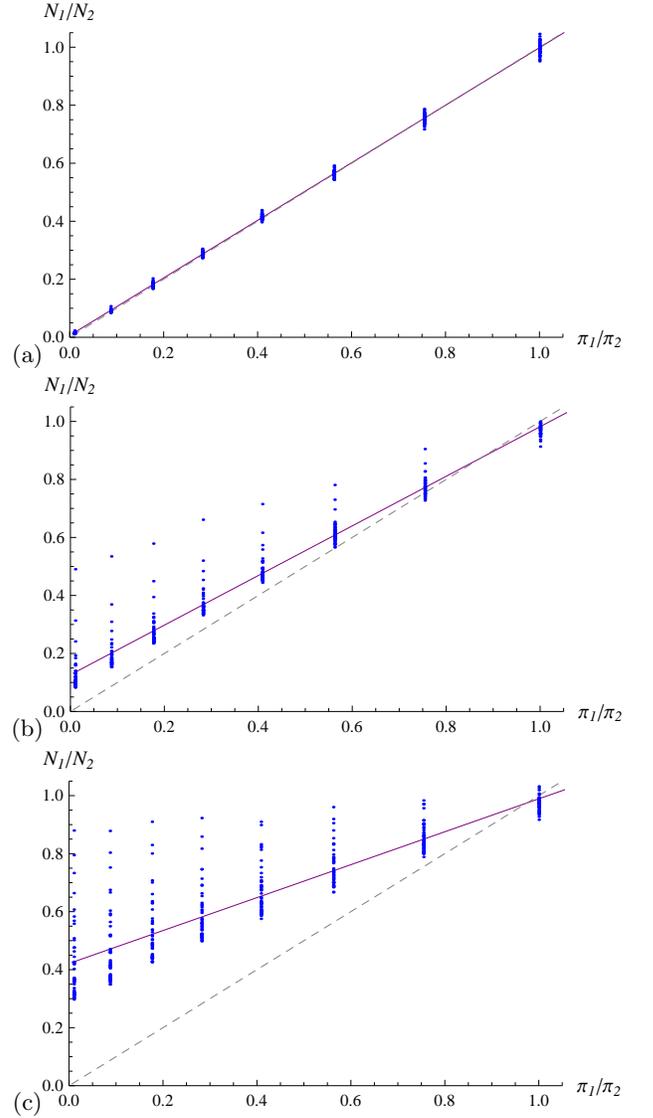


FIG. 12. **Output according to tailed distribution.** The plots in Fig. 12 (a), (b), and (c) show the ratios N_1/N_2 of the counts in the numerical simulations in comparison with the corresponding ratios π_1/π_2 according to the (tailed) stationary distribution, for error rates $\sigma = \pi/100$, $\sigma = \pi/20$, and $\sigma = \pi/10$, respectively. The solid purple lines show the best linear fits, which should match the 45° diagonal, shown as dashed gray line, in an ideal RPS agent. Each group of data points along a vertical line corresponds to fixed value of π_1/π_2 , but varying ϵ . The data used is in fact the same as that used for Fig. 9.

plemented in a laboratory as a proof-of-principle demonstration of learning agents enhanced by employing quantum physics.

that the output efficiency is bounded below 1, and depends on γ . For our examples we opt to consider the case where this efficiency is at 0.9.

-
- [1] D. Deutsch, *Proc. R. Soc. Lond. A* **400**, 97 (1985).
- [2] D. Deutsch and R. Jozsa, *Proc. R. Soc. Lond. A* **439**, 553 (1992).
- [3] P. Shor, in Foundations of Computer Science, *Proc. 35th Ann. Symp. Found. Comp. Sc.*, 124–134 (1994).
- [4] L. K. Grover, *Proc. 28th ACM Symp. Theory Comput. (STOC)*, 212–219 (1996) [arXiv:quant-ph/9605043].
- [5] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, U.K., 2000).
- [6] H. Neven, V. S. Denchev, G. Rose, and W. G. Macready, e-print arXiv:0811.0416 [quant-ph] (2008).
- [7] D. Manzano, M. Pawłowski, and Č. Brukner, *New J. Phys.* **11**, 113018 (2009) [arXiv:0904.4571].
- [8] E. Aïmeur, G. Brassard, and S. Gambs, *Mach. Learning* **90**, 261 (2013).
- [9] K. L. Pudenz and D. A. Lidar, *Quantum Inf. Process.* **12**, 2027 (2013) [arXiv:1109.0325].
- [10] S. Lloyd, M. Mohseni, and P. Rebentrost, e-print arXiv:1307.0411 [quant-ph] (2013).
- [11] G. D. Paparo, V. Dunjko, A. Makmal, M. A. Martín-Delgado, and H. J. Briegel, *Phys. Rev. X* **4**, 031002 (2014) [arXiv:1401.4997].
- [12] H. J. Briegel and G. De las Cuevas, *Sci. Rep.* **2**, 400 (2012) [arXiv:1104.3787].
- [13] M. Szegedy, in Foundations of Computer Science, *Proc. 45th IEEE Symp. Found. Comp. Sc.*, 32–41 (2004).
- [14] F. Magniez, A. Nayak, J. Roland, and M. Santha, *SIAM J. Comput.* **40**, 142 (2011) [arXiv:quant-ph/0608026].
- [15] J. Mautner, A. Makmal, D. Manzano, M. Tiersch, and H. J. Briegel, *New Generation Computing* (accepted, 2014) [arXiv:1305.1578].
- [16] J. T. Barreiro, M. Müller, P. Schindler, D. Nigg, T. Monz, M. Chwalla, M. Hennrich, C. F. Roos, P. Zoller, and R. Blatt, *Nature (London)* **470**, 486 (2011) [arXiv:1104.1146].
- [17] K. Mølmer and A. Sørensen, *Phys. Rev. Lett.* **82**, 1835 (1999).
- [18] N. Friis, V. Dunjko, W. Dür, and H. J. Briegel, *Phys. Rev. A* **89**, 030303(R) (2014) [arXiv:1401.8128].
- [19] M. Araújo, A. Feix, F. Costa, and Č. Brukner, e-print arXiv:1309.7976 [quant-ph] (2013).
- [20] J. Thompson, M. Gu, K. Modi, and V. Vedral, e-print arXiv:1310.2927 [quant-ph] (2013).
- [21] J. I. Cirac and P. Zoller, *Phys. Rev. Lett.* **74**, 4091 (1995).
- [22] F. Schmidt-Kaler, H. Häffner, M. Riebe, S. Gulde, G. P. T. Lancaster, T. Deuschle, C. Becher, C. F. Roos, J. Eschner, and R. Blatt, *Nature (London)* **422**, 408 (2003).
- [23] A. Y. Kitaev, in *Electr. Colloq. Comput. Compl. (ECCC)* **3** (1996) [arXiv:quant-ph/9511026].
- [24] A. A. Melnikov, A. Makmal, and H. J. Briegel, to appear in *Artificial Intelligence Research* (2014) [arXiv:1405.5459].