UNIVERSIDAD AUTONOMA DE MADRID

Escuela Politécnica Superior

22|23

Bachelor's Thesis

Long-range Quantum Information Transfer assisted by Machine Learning

Pablo Ernesto Soëtard García

Escuela Politécnica Superior
Universidad Autónoma de Madrid
C\Francisco Tomás y Valiente nº 11

Campus Internacional excelencia UAM CSIC+

www.uam.es

UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Degree in Telecommunications Technologies and Services Engineering

BACHELOR'S THESIS

Long-range Quantum Information Transfer assisted by Machine Learning

Author: Pablo Ernesto Soëtard García
Advisor: Gloria Platero Coello
Speaker: Alberto Suárez González

June 2023

Pablo Ernesto Soëtard García
Long-range Quantum Information Transfer assisted by Machine Learning

Pablo Ernesto Soëtard García
C\ Francisco Tomás y Valiente Nº 11

PRINTED IN SPAIN

A mi familia y amigos

We are trying to prove ourselves wrong as quickly as possible,
because only in that way can we find progress.

Richard P. Feynman

Agradecimientos

En primer lugar, me gustaría agradecer al Consejo Superior de Investigaciones Científicas (CSIC) por haberme dado la oportunidad de colaborar con ellos y por el apoyo económico percibido gracias a la beca JAE Intro otorgada.

Me gustaría agradecer el apoyo, consejos y ayuda indispensable brindada por David Fernández Fernández y mi tutora, Gloria Platero Coello, miembros del grupo en Nuevas Plataformas y Nanodispositivos para Computación y Simulación Cuántica en el Instituto de Ciencia de Materiales de Madrid (ICMM) del CSIC. Sin ellos este proyecto de investigación no habría sido posible.

Y por último a mis padres, que me han apoyado y animado durante el transcurso de toda la investigación y el duro camino para finalizar ambos grados, que culminan con este trabajo.

Resumen

Sabemos desde hace tiempo que los ordenadores cuánticos tienen el potencial de aportar un poder de cálculo sin precedentes, pueden arrojar luz en muchos campos distintos como el descubrimiento de nuevos fármacos, el desarrollo de nuevos protocolos de ciberseguridad y en el descubrimiento de nuevos materiales revolucionarios. Pero antes de poder conseguir todo esto, uno de los principales retos a los que se enfrentan los investigadores es la escalabilidad de estos ordenadores cuánticos hasta un tamaño útil. Debido a la naturaleza de los ordenadores cuánticos, no es tan fácil escalarlos como en el caso de los ordenadores clásicos, en los que basta con agregar más transistores para hacerlos más potentes. En cuanto a los ordenadores cuánticos, la adición de más cúbits generalmente implica que el sistema está más expuesto a los ruidos del entorno. Por lo tanto, en la actualizad sólo podemos obtener resultados significatios con sistemas con un reducido número de cúbits.

Una posible solución para este problema es la división del gran sistema cuántico en subsistemas cuánticos más pequeños. De este modo, los cálculos seguirían estando en nodos pequeños que serían más estables que el sistema agregado. Para que esta nueva arquitectura sea factible, se deben conseguir comunicaciones de largo alcance en el chip entre los nodos.

En este trabajo, proponemos un novedoso enfoque para estudiar la generación de nuevos protocolos que permitan la transferencia de información cuántica entre los nodos a distancias arbitrarias. Estos protocolos se basan en pulsos electromagnéticos que se aplican al sistema para transferir los cúbits al destino deseado. Ya existen protocolos en la literatura que realizan transferencias de información a distancias arbitrariamente largas, pero sólo funcionan en sistemas ideales, sin ninguna de las perturbaciones de ruido que surgen en una configuración real. Analizamos el uso de una nueva arquitectura basada en Aprendizaje por Refuerzo Profundo y agentes continuos para encontrar los pulsos de protocolos óptimos para un sistema concreto. Mediante conjeturas se navega por el vasto hiperespacio de combinaciones de pulsos. Además, nuestra arquitectura está diseñada con los efectos del ruido en mente, siendo capaz de entrenar nuestros modelos teniéndolo en cuenta, haciéndolos adecuados para generar protocolos robustos al ruido. Por otro lado, nuestra arquitectura es extensible a sistemas más grandes, en los que los protocolos más avanzados no consiguen una transferencia óptima.

Palabras clave

Transferencia de Información Cuántica, Aprendizaje por Refuerzo Profundo, Agentes Continuos, Protocolos robustos al ruido, Puntos cuánticos semiconductores

Abstract

We have known for a long time that quantum computers have the potential of bringing unprecedented computational power, they can throw light in many distant fields such as the discovery of new drugs, the development of new cybersecurity protocols and in the discovery of new revolutionary materials. But before we can achieve all of this, one of the main challenges that researches are facing is the scalability of those quantum computers to a useful size. Due to the nature of quantum computers, it is not as easy to scale them as with classical computers, where simply aggregating more transistors make them more powerful. In the case of quantum computers, the addition of more qubits generally implies that the system is more exposed to the environment noises. Therefore, only with systems formed by a reduced number of qubits can we currently achieve meaningful results.

A possible solution for this problem is the division of the large quantum system in smaller quantum sub-systems. In that way, the computations would still remain in small nodes that will be more stable than the aggregated system. For this new architecture to be feasible, long range on-chip communications between the nodes should be attained.

In this work, we propose a new approach to study the generation of new protocols that allow for quantum information transfer between smaller nodes at an arbitrary distance. This protocols are based on electromagnetic pulses that are applied to the system to transfer the qubits to the desired destination. There already exist state-of-the-art protocols that perform arbitrarily long information transfers, but they only work on ideal systems, without any of the noise disturbances that arise on a real setup. We analyze the use of a new architecture based on Deep Reinforcement Learning and continuous agents to find the optimal protocol pulses for a particular system by making educated guesses while surfing the vast hyperspace of combinations of pulses. Furthermore, our architecture is designed with noise effects in mind, being able to noisy-train our models, making them suitable to generate noise-robust protocols. In addition, our architecture is extensible to larger systems where state-of-the-art protocols do not achieve an optimal transfer.

Keywords

Quantum Information Transfer, Deep Reinforcement Learning, Continuous Agents, Noise-robust protocols, Semiconductor quantum dots

Table of Contents

Lists

List of equations

List of figures

List of tables

1

Introduction

This work combines two of the most promising technologies of our times, namely quantum computing and Artificial Intelligence (AI) . Both fields have historically involved a lot of expert knowledge. In recent years the power of AI has become mainstream due to its optimisation capabilities and it has started to be applied in very distant fields, such as quantum computing [1–4]. In this work a subset of AI called Machine Learning (ML) , specifically a set of algorithms that belong to the Reinforcement Learning (RL) family has been used in the identification of new long-range quantum information transfer protocols. These are protocols to transfer information at large distances, with minimal population in the intermediate sites of the quantum dots chains. In order to introduce the unfamiliar reader to those complex topics, a brief introduction on the key concepts of each of the disciplines will follow.

1.1 Brief Introduction to Quantum Phenomena

The quantum description of the physical world was developed at the beginning of the 20th century. Since then it has led to multiple advances in our understanding of the universe, leading to one of the most important theories of science, the standard model. This model describes the fundamental blocks that build up the whole universe, particles. This phenomena has also fostered technology, allowing to create more accurate and sensitive sensors and actuators as well as potentially opening the way to a whole new computing paradigm, Quantum Computing. For a more in depth introduction on the topic see [5].

1.1.1 Quantum Mechanics

Lets start by introducing some key concepts of quantum mechanics that are essential to understand the basis on which this work has been developed. Quantum mechanics' mathematical framework is well defined and based on a set of four postulates, which can be found in appendix A.

Wave Functions, Hamiltonians and Measurements

In the context of quantum mechanics, a wave function $|\psi\rangle$, is a complex-valued function which serves as a mathematical description of a quantum system. It contains all the information about the system's quantum state. The absolute square of the wave function, $||\psi\rangle|^2$, gives the probability density of finding the particle at a particular location in space at a given time. In this work, these probabilities are called *population*.

In quantum mechanics, the total energy of a system is given by the Hamiltonian operator H. The eigenstates of the Hamiltonian are the stationary states of the system. Each eigenstate $|\alpha\rangle$ has a corresponding eigenenergy E_α, such that $H|\alpha\rangle = E_\alpha|\alpha\rangle$.

An eigenstate is a particular state of the system that, when subjected to a measurement, yields a definite value for a specific observable. The eigenenergies represent the possible energy values that a system can have. When a measurement of the energy is performed on the system in an eigenstate $|\alpha\rangle$, the result is always its eigenenergy E_α. In this work we will investigate the eigenstates in the context of positions of a chain of artificial atoms (quantum dots) and their spatial distribution. Their corresponding eigenenergies are the energies associated with each state.

Measuring a quantum system refers to the process of extracting information about one or more observable properties of the system, in our case, positional information. When a measurement is performed on a quantum system, the outcome is probabilistic, given by $||\psi\rangle|^2$. The result of a measurement is one of the possible eigenvalues associated with the observable being measured, and the system is projected into the corresponding eigenstate. When a wave function is measured it collapses, this means that it suffers an irreversible change upon measurement, losing some state information. After the collapse, the system is in the eigenstate corresponding to the measured value, and further measurements of the same observable will yield the same result. The collapse is often interpreted as a transition from a superposition of possible states to a definite state.

The Schrödinger Equation

Wave functions can evolve over time, and their behavior is governed by the Schrödinger equation, which is a fundamental result in quantum mechanics and provides a mathematical framework to describe the behavior of quantum systems. The Schrödinger equation is a partial differential equation that describes how the wave function evolves in time and how it responds to external potentials. It allows us to calculate the probability distribution of the system's properties and predict the probabilistic outcomes of measurements. Mathematically, the time-dependent Schrödinger equation is defined as:

$$i\hbar\frac{\partial|\psi\rangle}{\partial t} = H|\psi\rangle \ , \tag{1.1}$$

where $\hbar$ is the reduced Planck constant that we are going to assume for the rest of the work to be $\hbar =$

1, i is the imaginary unit, ψ is the wave function, t is time, and H is the Hamiltonian operator, which can also be time-dependent. The Schrödinger equation is a vectorial one-way wave equation that essentially states that the rate of change of the wave function with respect to time is proportional to the Hamiltonian acting on the wave function. Solving the Schrödinger equation yields the time evolution of the wave function, allowing us to determine how it changes over time and how the quantum system behaves. We can make predictions about its observable properties through the calculation of expectation values or probabilities.

Pure vs Mixed Systems, Density Matrices

A pure quantum system refers to a system that is described by a wave function that is in a well-defined state, like the ones previously discussed. The wave function $|\psi\rangle$ associated with the pure system is a complete description of it, capturing all the information about its quantum state. It can evolve over time, and the evolution of the system would remain within the space of pure states, preserving the purity of the system.

In contrast, a mixed state refers to a statistical ensemble or a mixture of pure states. In a mixed state, the system is in a probabilistic combination of different pure states. Mixed states arise when there is uncertainty or lack of knowledge about the actual state of the system, such as in situations involving interactions with the external environment of the system.

In these kind of systems, instead of working with wave functions and the Schrödinger equation to solve their evolution over time, we will work with density matrices and master equations to solve the system. The density matrix, denoted by ρ, is a mathematical representation of the state of a mixed quantum system. The elements of the density matrix ρ can be interpreted as the probabilities and coherences between different states of the system. The diagonal elements represent the probabilities of finding the system in the corresponding state, while the off-diagonal elements represent the coherences or interference terms between different states. Mathematically, a density matrix is a Hermitian, positive semi-definite matrix with unit trace, which is constructed applying equation:

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \ , \tag{1.2}$$

where p_i represents the probability of the system being in the pure state $|\psi_i\rangle$.

The Master Equation for Mixed Systems

The master equations, also known as the Liouville-von Neumann equations, are a set of differential equations that describe the time evolution of the density matrix of an open quantum system. They provide a way to model and analyze the dynamics of quantum systems that interact with their environment. Mathematically, the master equation is given by equation:

$$\frac{\partial \rho}{\partial t} = -i[H, \rho] + \sum_k L_k(\rho) \ , \tag{1.3}$$

where $\frac{\partial \rho}{\partial t}$ represents the time derivative of the density matrix, H is the system's Hamiltonian operator, $[H, \rho]$ is the commutator of H and ρ, and $\sum_k L_k(\rho)$ represents the Lindblad operators that describe the system-environment interactions [6]. The term $-i[H, \rho]$ represents the unitary evolution of the system governed by its Hamiltonian. It describes how the system would evolve in isolation, without any interactions with the environment. The term $\sum_k L_k(\rho)$ accounts for the effects of the system's interactions with its environment. The Lindblad operators are defined as $L_k(\rho) = \gamma_k \left(L_k \rho L_k^\dagger - \frac{1}{2}\{L_k^\dagger L_k, \rho\} \right)$ where γ_k is the damping rate and $\{L_k^\dagger L_k, \rho\}$ is the anticommutator operator, where the $\dagger$ operator means the transpose and complex conjugate. These operators introduce dissipation, decoherence, and noise into the system, leading to irreversible processes. They are typically chosen to model specific types of interactions, such as energy exchange, relaxation, or dephasing.

Solving the master equation provides the time-dependent evolution of the density matrix, allowing us to study the dynamics of open quantum systems to analyze the behavior of the system in realistic environments. They provide a powerful tool for investigating the impact of noise and decoherence on the quantum properties of a quantum system, this is why they are applied multiple times in this work.

1.1.2 Quantum Computers

Quantum computers are a type of computer that make use of quantum mechanics phenomena to process information. The idea of quantum computing was first introduced in the early 1980s by Richard Feynman [7], who proposed that quantum computers could be used to simulate the behavior of quantum systems that are too complex for classical computers to handle. The potential applications of quantum computers are vast and include fields such as cryptography [8], machine learning [9], and drug discovery [10].

Researchers have made significant progress in developing the technology necessary to build a practical quantum computer, what is known as the quantum supremacy [11]. However, quantum computers are still in their early stages of development and face significant technical challenges, we are still in the Noisy Intermediate Scale Quantum computers era. This work tackles one of them, scalability by information transfer inside this type of computers. A fast and reliable way of connecting quantum processing modules would potentially allow them to be scaled. But before discussing that, lets start by explaining the fundamental blocks of quantum computers.

Qubits vs Bits

A qubit, or quantum bit, is the basic unit of quantum information, analogous to a classical bit in classical computing. However, unlike classical bits, which can only have two possible values (0 or 1), a qubit

can exist in a superposition of states. This means that a single qubit can represent multiple values simultaneously, which allows for parallel computation and faster processing. For example, a qubit could be in a state were it has the same weight of 0 and 1, its wave function would be $|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. Remember that in section 1.1.1 we discussed that the wave function describes a probability distribution, in our two-level system, the eigenstates are $|0\rangle$ and $|1\rangle$, therefore its wave function is $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha^2 + \beta^2 = 1$ to fulfil the property of a normalized probability distribution, that is why there is a factor of $\frac{1}{\sqrt{2}}$. Having that in mind, the fact that $\alpha = \beta = \frac{1}{\sqrt{2}}$ implies that the probability of having our system in state $|0\rangle$ is equal to the one of having it in state $|1\rangle$, which are equal to $\left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2} = 50\%$.

The state of a qubit can be represented using the Bloch sphere, which is a graphical representation of the qubit's state. The Bloch sphere is a three-dimensional sphere, where the poles represent the states $|0\rangle$ and $|1\rangle$, and all other points represent superpositions of those states. The state of a qubit can be visualized as a unit vector over the surface of the Bloch sphere, as show in figure 1.1, where the relative phase between states is given by the azimuthal angle φ and the probability amplitudes of the states by the polar angle θ.

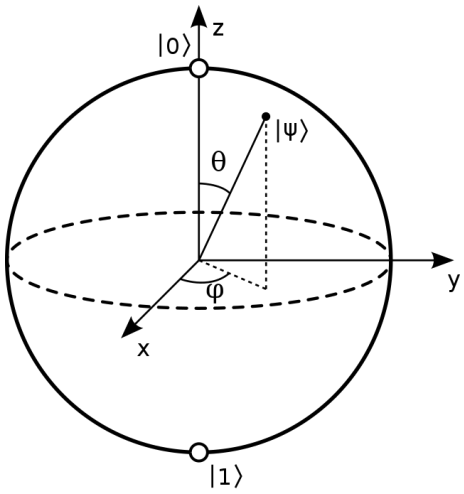

Figure 1.1: Bloch Sphere. From wikipedia. A qubit state $|\psi\rangle$ can be represent in it, where the relative phase between states is given by the φ angle and the probability amplitudes of the states by the θ angle.

Additionally, multiple qubits can exhibit entanglement between each other, like in $|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$, which means that the state of one qubit is linked to the state of another qubit, regardless of their physical separation. This property can be used for secure quantum communication and distributed computing.

Further discussion on qubit operators and quantum gates can be found in appendix B.

Classic vs Quantum Computers

Thanks to the properties of quantum systems such as qubits, we can perform multiple operation in *"parallel"*. This is true in some sense, but in order to be completely true we would need to be able to extract all those results of the parallel computations at once, and that is not possible, we can only retrieve one of them. Due to the collapse of the wave functions, once we measure, the remaining information is lost. You may be wondering why is that the case, and the answer is that all the results are in superposition, and thus they have a probability associated with them. When measuring the system we will only get one result, the odds of getting one or another is related by their associated probabilities (like in the case explained in 1.1.2, on the superposition of states $|0\rangle$ and $|1\rangle$, we would get one or the other with a 50% chance). Therefore we would need to wisely manipulate those results in order to make the one that we want to retrieve as likely as possible. By doing this we could perform "parallel" computations in quantum systems, and potentially get the desired result exponentially faster than in classical computers, which would need to execute each computation sequentially. In fact for each extra qubit we add to our algorithm we double the computational power of our system, in a classical setup this power would increase linearly with the amount of bits. This is translated in a computational power of $O(2^n)$ using qubits vs $O(n)$ using bits, being n the number of qubits and bits respectively. For readers interested on the topic who really want to appreciate the power of this computational paradigm, a simple quantum algorithm, the *Deutsch-Jozsa algorithm* [12], is explained in appendix C.

The power of this type of *"parallel"* computations has been materialized in multiple other algorithms like *Grover's algorithm*, a search algorithm that has a time complexity of $O(\sqrt{N})$ instead of the classical $O(N)$. Another famous quantum algorithm is *Shor's algorithm* [8], which efficiently factors large numbers, a problem that is believed to be intractable for classical computers. If implemented in large quantum computers it would be able to break current encryption schemes like *RSA*.

Other promising application of quantum computers is Quantum AI. We can see in the *Deutsch-Jozsa* algorithm, explained in appendix C, that we can evaluate a function in all its span at once. This could be used in optimization problems, to find the minimum (or maximum) of a function just calling it once. In classical computers the cost function of the Neural Network (NN) used in AI is optimized during training, taking (usually) a gradient descent approach, that does not guarantee to fall into an absolute minimum. This would not be the case with quantum computers, since we could potentially find the minimum of the cost function with just one evaluation of it, training the NN in just one pass. We will not focus on this type of AI in this work, we are interested in applying classical machine learning algorithms to quantum computing, not using quantum Machine Learning.

Qubit Platforms

There exists a wide variety of platforms that could potentially be used as qubit holders and manipulators. In this work we are only going to discuss two of them: superconducting qubits and semiconductor spin

qubits. The former is the most prevalent platform in the industry, while the latter is going to be the platform that this work will focus on.

- Superconducting qubits are a type of qubit that rely on the properties of superconducting circuits to create a two-level quantum system. They are widely used in the industry and are the basis of the quantum computers of companies like Google [11] or IBM [13]. They are easy to fabricate and control, and have relatively long coherence times, which means they can maintain their quantum states for long enough periods to allow for meaningful computations.
- Semiconductor spin qubits are a type of qubit that rely on the implementation of discrete two-level quantum systems in a semiconductor-based device, such as a quantum dot. The discrete systems can be manipulated and read using electromagnetic fields, allowing for quantum operations. Semiconductors are well-established materials in the chip-manufacturing industry, meaning that semiconductor spin qubits can be fabricated using existing technologies. Furthermore, semiconductor spin qubits have demonstrated long coherence times (e.g. silicon can be purified to reduce hyperfine interactions [14], which are the main cause of decoherence), making them promising candidates for practical quantum computing applications.

Main challenges

Quantum computers face several significant challenges that need to be addressed before they can be used in practical applications. We currently have Noisy Intermediate-Scale Quantum (NISQ) computers, which are a type of quantum computers that operate with a relatively small number of qubits, usually between 50 and a few hundred. The term *"Noisy"* refers to the fact that NISQ devices have a relatively high error rate due to factors such as environmental and system noise that cause decoherence and imperfect control of the qubits. They are *"Intermediate-Scale"* because they lay between small-scale quantum devices that can be simulated on classical computers and large-scale fault-tolerant quantum computers that can perform error-corrected quantum computations. Some of the main challenges that have to be addressed before achieving large-scale fault-tolerant universal quantum computers include:

- Error correction: Quantum computers are highly sensitive to noise and errors, which can cause decoherence and lead to incorrect results. Developing efficient and reliable error correction methods is crucial for the development of practical quantum computers.
- Scalability: Quantum computers must be able to scale to large numbers of qubits to perform useful computations. However, as the number of qubits increases, so does the complexity of controlling and measuring them, which presents significant engineering challenges.
- Interconnectivity: Quantum computers must be able to communicate and exchange information with each other to perform distributed computations. Developing efficient and secure methods for quantum communication is a major challenge.
- Hardware limitations: The current generation of quantum computers faces limitations in terms of coherence times, gate fidelities, and qubit connectivity. Improvements in hardware design and fabrication are necessary to address these limitations.
- Algorithm design: Developing efficient and useful quantum algorithms for a variety of applications is still an active area of research, and many problems remain open.
- Practical applications: Identifying practical applications of quantum computing that can outperform classical computers is a significant challenge, and many of the most promising applications are still in the early stages of development.

Despite their limitations, NISQ computers are seen as an important step in the development of practical quantum computers. Researchers are actively working to improve the performance of NISQ devices and develop new algorithms that can take advantage of their capabilities, with the ultimate goal of developing large-scale fault-tolerant quantum computers capable of outperforming classical computers for a wide range of applications.

1.1.3 Quantum Information Transfer

A key milestone that would be crucial in solving most of the challenges stated previously would be to be able to achieve reliable and robust quantum information transfers. In classical computers, information transfer happens by copying the information (bit) at the source and sending it over a medium such as a copper trace, a wire, optic fiber, air... using electromagnetic fields. Unfortunately, in the quantum world there exist the **no-cloning theorem**, which is a fundamental result in quantum mechanics that states that it is impossible to create an exact copy of an unknown quantum state. This means that there is no way to duplicate a quantum state perfectly, as this would violate the laws of quantum mechanics. When a quantum state is measured or observed, the wave function collapses and the original state is lost. Therefore, any attempt to copy a quantum state will necessarily disturb the original state, leading to an imperfect copy.

But, quantum information transfer is necessary to communicate different quantum modules and achieve modular and scalable quantum computers, so, is this the end? Luckily no, we can take advantage of the quantum phenomena to achieve our goal. We have discussed that the evolution of a quantum system is represented by its wave function, that means that there is a non-zero probability to measure the particle in the end of the quantum dot array (following the evolution imposed by the *Schrödinger equation*). If we are able to tweak the Hamiltonian of our system to guide our population to the desired location we could accomplish quantum information transfer. In this case, instead of copying the information, we would be directly transferring it to our desired destination.

Hamiltonian Control

As described in subsection 1.1.1, having a Hamiltonian that describes the evolution of a quantum system, its diagonal elements (Δ_i) describe the energy differences between the eigenenergies of the system. On the other hand, the off-diagonal elements (Ω_{ij}), which are the coupling factors, are the terms that describe the interactions between the states of the system. These interactions can be caused by various physical processes, such as the interaction of the system with electric and magnetic fields. An example of the control Hamiltonian [15] of a three-level system written in basis $\{|1\rangle, |2\rangle, |3\rangle\}$ is represented in equation (1.4), and a schematic picture of the system is illustrated in figure 1.2. In this example we assume that only adjacent states interact with one another, and that the energy origin is on the third state. Note that this Hamiltonian is given for a single spinless fermionic particle. Therefore,

we have a total of three states, instead of the six-level Hamiltonian which would result from a spin half particle such as an electron.

$$H = \begin{bmatrix} \Delta_1 & -\Omega_{12} & 0 \\ -\Omega_{12} & \Delta_2 & -\Omega_{23} \\ 0 & -\Omega_{23} & 0 \end{bmatrix} . \tag{1.4}$$

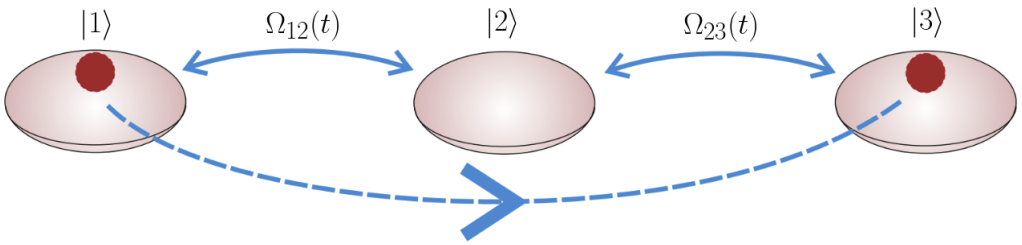

Figure 1.2: Three-level information transfer scheme. Adapted from [16]. Information transfer from $|1\rangle$ to $|2\rangle$ is modulated by Ω_{12} and from $|2\rangle$ to $|3\rangle$ by Ω_{23}. Tuning the coupling between states, the particle is directly transferred from site $|1\rangle$ to site $|3\rangle$, with minimal population in the intermediate site.

The interactions between the states of the Hamiltonian caused by the coupling elements can have important implications for the behavior of the system. For example, they can lead to the transfer of energy or information between different parts of the system, or they can cause the system to evolve over time in complex ways.

The goal is to apply specially modulated electric fields, using electrical pulses to the system in order to tweak the coupling factors of its Hamiltonian and accomplish the desired population transfer. In this case targeted to a system of quantum dots coupled by the tunnel effect. Each one would represent one of the states and the Ω_{ij} would be the couplings caused by the tunnels between them.

Follow the Dark States

If the driving of the Hamiltonian is slow enough, the system will evolve following their instantaneous eigenstates (adiabatic theorem, as shown bellow in equation (1.8)). In systems with an odd number of spacial states, it can be proved that there exist eigenstates that only populate the odd states, those are the so called dark states since they do not let even states, or in our case, states in even positions, to be populated. From the Hamiltonian described in equation (1.4), and assuming $\Delta_1 = 0$ (this is a mandatory condition for the existence of a dark state), we can calculate its eignestates and confirm that indeed there exist a dark state $|D_0\rangle$ that does not populate the state $|2\rangle$ (and therefore the central dot), as seen in the following equations:

$$
\begin{aligned}
|D_+\rangle &= \sin\Theta_1 \sin\Theta_2 |1\rangle + \cos\Theta_2 |2\rangle + \cos\Theta_1 \sin\Theta_2 |3\rangle \ , \\
|D_-\rangle &= \sin\Theta_1 \cos\Theta_2 |1\rangle - \sin\Theta_2 |2\rangle + \cos\Theta_1 \cos\Theta_2 |3\rangle \ , \\
|D_0\rangle &= \cos\Theta_1 |1\rangle - \sin\Theta_1 |3\rangle \ ,
\end{aligned}
\tag{1.5}
$$

where:

$$
\begin{aligned}
\Theta_1 &= \arctan(\Omega_{12}/\Omega_{23}) \ , \\
\Theta_2 &= \frac{1}{2} \arctan[(\sqrt{(2\Omega_{12})^2 + (2\Omega_{23})^2})/\Delta_2] \ .
\end{aligned}
\tag{1.6}
$$

Dark states can be used to protect quantum information, by restricting the amount of interactions of the system with the environment. If a state is not populated, it cannot be disturbed by the environment, thus avoiding unwanted effects like decoherence or loss of population. Therefore, if we manage to modulate the electric fields, so that the Hamiltonian of the system follows a dark state, systems with an odd number of states would be preferable than even ones.

The next challenge is to find the pulses Ω_{12} and Ω_{23} that satisfy the equations (1.5) and (1.6). A naive approach can be to apply two linear pulses, one ascending (Ω_{12}) and the other one descending (Ω_{23}). At first glance this could seem a good idea, since we are first allowing the population to move from $|1\rangle$ to $|2\rangle$ and then from $|2\rangle$ to $|3\rangle$. But as it can be seen in figure 1.3, this intuitive approach is not suitable for transferring the population to $|3\rangle$. One could then start analyzing equations (1.5) and (1.6). We want our system to follow the dark state $|D_0\rangle$, and that at $t = 0$ our population should be in state $|1\rangle$ and at $t = t_{max}$ it should be on state $|3\rangle$, this are the so called boundary conditions. If we meet that criterion we would be able to claim that we have successfully achieve information transfer from $|1\rangle$ to $|3\rangle$. All of this implies that we should have $\Theta_1 = 0$ to meet $|D_0\rangle = \cos\Theta_1|1\rangle - \sin\Theta_1|3\rangle = |1\rangle$. If $0 = \Theta_1 = \arctan(\Omega_{12}/\Omega_{23})$ then $\Omega_{12}/\Omega_{23} = 0$. This means that at $t = 0$, $\Omega_{23} \gg \Omega_{12}$. An analogous derivation can be made for $t = t_{max}$, where $\Omega_{23} \ll \Omega_{12}$ should be met. A protocol that meets those conditions could be one having $\Omega_{12} = 0, \Omega_{23} = \Omega_0$ at $t = 0$, and $\Omega_{12} = \Omega_0, \Omega_{23} = 0$ at $t = t_{max}$, where Ω_0 is the maximum tunneling rate, and will be our normalization parameter across the hole work. Then we could linearly interpolate those start and end points and we would get our second less naive protocol, that is the inverse of our first protocol guess. As it can be seen in figure 1.3, with this simple counter-intuitive protocol derived from our analysis, we manage to successfully transfer the population from state $|1\rangle$ to $|3\rangle$.

Transfer Velocity vs Fidelity

Effects like decoherence or dephasing caused naturally in the system and also due to its interaction with the environment produce information loss across time, lowering the overall fidelity of the transfer. That is why we need quantum information transfer protocols that are fast. On the other hand, we want a protocol with a high fidelity, meaning that as much of the system's population as possible arrives to the target state at the end of the protocol, in order to have a higher probability of successful information

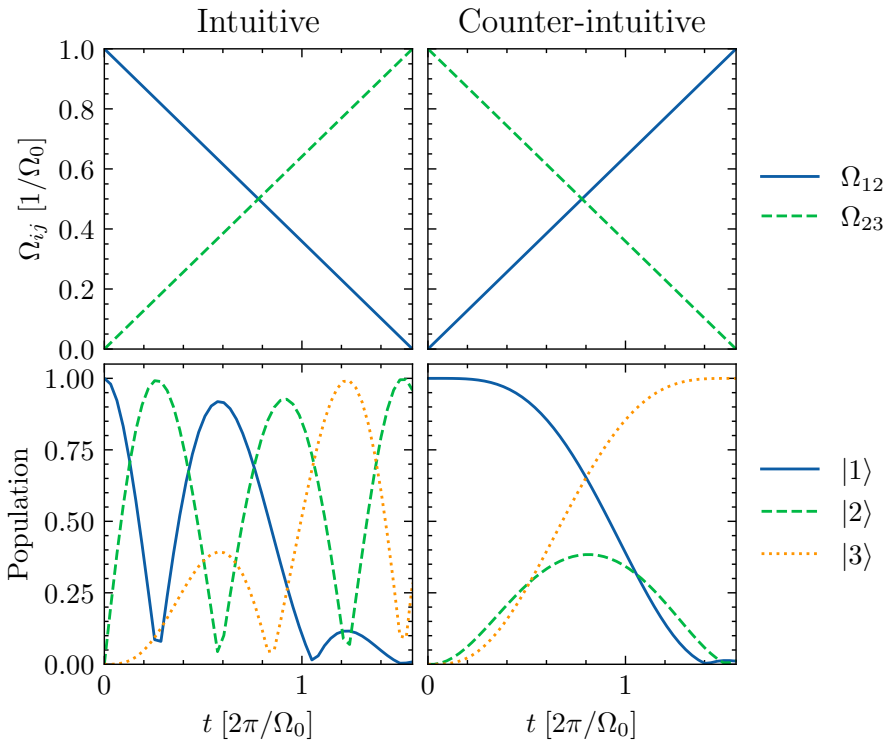

Figure 1.3: Intuitive vs Counter-intuitive Protocols. One scenario per column: intuitive (left), counter-intuitive (right). First row represents the protocol's pulses. Second row shows the dynamics that these protocols induce on the populations of a three-level system initialized at state $|1\rangle$, with $\Delta_i = 0$.

transfer. In this work we define fidelity as the projection or expectation of the final quantum state on the target state:

$$\mathcal{F} = |\langle N|\psi(t_{max})\rangle|^2 \,, \tag{1.7}$$

where N is the length of the quantum dot chain. Having a maximum of $\mathcal{F} = 1$ when the final and target states are the same. All these factors have to be taken into account while designing the pulses that will drive the electromagnetic fields applied to the system, that at the same time, will control its Hamiltonian's evolution.

1.2 Introduction to Machine Learning

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that focuses on developing algorithms and statistical models that enable computer systems to learn from data. By learning from data, ML algorithms can generalize to new, unseen data, and make predictions or decisions based on it. This property allows machine learning algorithms to be used in a wide variety of applications, such as classification, image recognition, natural language processing, recommendation systems and many others. With the increasing availability of data and computational resources, the potential for machine learning to solve complex problems and make a significant impact on society is constantly growing.

One of the most important properties of this models is that they can act as universal function approximators, meaning that they can approximate any function, given enough data, computational resources and model size. This property is the one that is going to be used in this work to model the electromagnetic pulses needed to drive the control Hamiltonian, as explained in 1.1.3.

1.2.1 Reinforcement Learning

Reinforcement Leaning (RL) is a subfield of ML, it deals with the problem of how an agent can learn to make decisions by interacting with a dynamic environment. In RL, an agent learns by receiving feedback from the environment in the form of rewards or punishments, based on the actions it takes. The objective of the agent is to learn an optimal policy that maximizes the cumulative reward over time, performing the task that it has been requested in the best way it has found.

The key concepts in RL are the following:

- Agent: Entity composed (generally) by neural networks that interacts with the environment. It observes the current state of the environment, selects an action based on a policy, and receives feedback from the environment in the form of a reward.
- Agent's Networks: Depending on the architecture one may encounter different types of networks, but generally speaking the agent is composed by two main ones:
    - Actor Network: A neural network that learns to select actions in a given state. It takes the current state as input and produces a probability distribution over possible actions that can be taken in that state. The actor network aims to maximize the expected cumulative reward by selecting the best action in each state.
    - Critic Network: A neural network that evaluates the quality of the action taken by the actor network. It takes the current state and the action taken by the actor as input and outputs a scalar value that represents the expected long-term reward from that state and action. The critic network helps the actor network to learn by providing feedback on the quality of its actions.
- Action Space: Set of all possible actions that the agent can take in the environment. In our case the amplitudes of the pulses at each step.
- Observation Space: Set of all possible states that the agent can observe in the environment. In our case the quantum system's wave function.

- Environment: External world in which the agent operates. It can be a physical system or a simulation. In our case it is a quantum system simulation.
- Cost Function and Reward: Scalar signal that the agent receives from the environment after taking an action. It indicates the quality of the action taken by the agent, with higher rewards indicating better actions. The environment provides feedback to the agent based on the actions it takes. This reward is calculated using a cost function that has to be carefully designed to attain maximum learning performance.
- Policy: Strategy that the agent uses to select actions based on the current state of the environment and past experiences interacting with the environment.
- Replay Buffer: Mechanism for storing and retrieving past experiences or transitions from an agent's interactions with its environment. It allows the agent to interact less with the environment by learning from past interactions.

Reinforcement learning algorithms use these concepts to iteratively learn an optimal policy by exploring the environment, selecting actions, and receiving feedback in the form of rewards. Over time, the agent improves its policy by maximizing the cumulative reward it receives from the environment.

1.3 State of the Art

Quantum information transfer protocols are an essential key component to achieve error corrected quantum computers. These protocols enable the transfer of quantum information between distant sites of a system, making it possible to build up large-scale quantum computing. Over the past few years, there has been significant progress in the development of new quantum information transfer protocols, with researchers exploring a range of different approaches and techniques.

Some of the most promising quantum information transfer protocols include sequential transfer, Coherent Transport by Adiabatic Passage (CTAP) [15], Shortcuts to Adiabaticity (STA) [16], and pulse engineering methods such as the Chopped RAndom Basis (CRAB) [17] and GRadient Ascent Pulse Engineering (GRAPE) [18] algorithms. Each of these protocols has its advantages and disadvantages, and researchers are actively exploring ways to optimize and improve them for a range of applications.

As the field of quantum information transfer continues to evolve, researchers are focused on developing new techniques that can overcome the limitations of existing protocols, including the effects of decoherence, noise, and environmental factors.

1.3.1 Sequential Transfer

One of the ways of achieving quantum information transfer between two states is to sequentially transfer the population through the intermediate states that conform the quantum dot chain, one site at a time. An example of such a protocol is applying a set of pulses with constant amplitudes to all the control gates of the system, when the area under the pulses reaches π, the sequential transfer between two adjacent dots is achieved. This approach is denominated a π-*pulse* protocol [19].

One of the main advantages of sequential transfer is its scalability [20]. The protocol can be used to transfer quantum states over long distances, by using a chain of intermediate sites to bridge the gap between the source and target dots.

However, one of the most important disadvantages of sequential transfer is its susceptibility to errors and decoherence. The transfer of quantum information through a chain of intermediate sites can be affected by a range of environmental factors, including noise, coupling to the environment, and imperfections in the control hardware. These factors introduce errors into the transfer process and limit the fidelity of the transferred state.

1.3.2 Ansatz Pulses

Defining a parameterized ansatz pulse is a technique used in quantum information transfer protocols to generate a set of effective control pulses. The term "ansatz" refers to a set of candidate solutions,

or trial functions, that satisfy certain physical constraints, such as being smooth and bounded. These pulses are then parameterized to generate a family of potentially effective control pulses.

One property of ansatz pulse methods is their ability to generate different pulse shapes varying their parametrization. This makes them useful for designing control pulses for complex quantum systems with many states, where a brute-force search of the entire pulse space may be impractical or impossible. Indeed this reduces the set of possible shapes, but still makes it challenging to find the proper parametrization of the pulses for a complex system (high dimensional Hilbert space). In contrast, for smaller systems, the effects of each of the parameters can be studied analytically.

However, one potential disadvantage of ansatz pulse methods is that they rely on the choice of an appropriate ansatz pulse or set of pulses. This makes them likely to not generate the most effective pulses for the particular quantum system where quantum information transfer should occur.

Coherent Transport by Adiabatic Passage (CTAP)

The CTAP protocol [15], uses two counter-intuitive Gaussian pulses to perform the quantum information transfer between the first and third states without populating the intermediate. This is only true if the adiabaticity condition is met. The criterion for adiabaticity is that the rate of change of the Hamiltonian must be slow compared to the energy gap between the eigenstate $|N(t)\rangle$ and the other eigenstates $|M(t)\rangle$ of the Hamiltonian $H(t)$, then the system will remain close to the eigenstate $|N(t)\rangle$ for all subsequent times:

$$|\langle M(t)|dH(t)/dt|N(t)\rangle| \ll |E_M(t) - E_N(t)|,\ \forall M \neq N\ . \tag{1.8}$$

That is the main disadvantage of this protocol, the system must evolve in a way that is adiabatic (slow) enough to avoid exciting unwanted transitions between energy levels. In this case, following the dark state that we discussed in 1.1.3 by making use of the pulses Ω_{12} and Ω_{23}, defined as:

$$\begin{aligned} \Omega_{12}(t) &= \Omega_0 \exp\left[-\left(t - \frac{t_{max}+\sigma}{2}\right)^2 \Big/ (2\sigma^2)\right]\ , \\ \Omega_{23}(t) &= \Omega_0 \exp\left[-\left(t - \frac{t_{max}-\sigma}{2}\right)^2 \Big/ (2\sigma^2)\right]\ . \end{aligned} \tag{1.9}$$

In its original version, CTAP only works on a three-level quantum systems. However, it can be extended to larger quantum system with an odd number of levels via straddling pulses [15]. These are Gaussian pulses with high amplitudes applied to the intermediate states to minimize their populations, defined as:

$$\Omega_s(t) = \Omega_{s0} \exp\left[-\left(t - \frac{t_{max}}{2}\right)^2 \Big/ (2\sigma^2)\right]\ . \tag{1.10}$$

Shortcuts to Adiabaticity (STA)

The STA protocol [16] is a technique used in quantum systems that allows to speed up CTAP by evolving the Hamiltonian faster in regions where the excited states are further apart. This is done by using specially designed control ansatz pulses to manipulate the system in such a way that it follows the adiabatic evolution, but at a much faster rate, reducing the lower bound of the adiabatic constant at any given time. This is the main advantages of STA over CTAP, it allows for the same level of fidelity in a much shorter amount of time. It can be proven that STA is the most efficient pulse that can be achieved in an ideal three state system, however this may not be the case for real or larger systems.

The STA protocol pulses Ω_{12} and Ω_{23} are derived by imposing the following wave function, that in the end should follow the dark state:

$$|\psi(t)\rangle = \cos\chi\cos\eta|1\rangle - i\sin\eta|2\rangle - \sin\chi\cos\eta|3\rangle \ . \tag{1.11}$$

After solving the time-dependant Schrödinger equation using the above wave function, we derive the relations that Ω_{12} and Ω_{23} should satisfy with our ansatz functions χ and η:

$$\begin{aligned} \dot{\chi} &= \tan\eta(\Omega_{12}\sin\chi + \Omega_{23}\cos\chi) \ , \\ \dot{\eta} &= \Omega_{12}\cos\chi - \Omega_{23}\sin\chi \ . \end{aligned} \tag{1.12}$$

A set of smooth pulses that fulfil the boundary conditions of the system can be derived by making:

$$\begin{aligned} \chi &= \frac{\pi t}{2t_{max}} - \frac{1}{3}\sin\left(\frac{2\pi t}{t_{max}}\right) + \frac{1}{24}\sin\left(\frac{4\pi t}{t_{max}}\right) \ , \\ \eta &= \arctan(\dot{\chi}/\alpha_0) \ , \end{aligned} \tag{1.13}$$

where χ is an ansatz weighted sum, whose coefficients could be tuned to achieve better information transfer protocols, and α_0 is an arbitrary parameter that controls the strength of the pulses and the total population of the intermediate state.

1.3.3 Gradient based methods

Gradient-based methods are a type of optimization techniques that are commonly used in quantum information transfer protocols to optimize a set of initial control pulses for quantum gates [21, 22] or state transfers [17, 18]. These methods work by iteratively adjusting the control pulses based on the gradient of a cost function, in order to find the optimal solution. The cost function used in these methods is typically a measure of the distance or difference between the target quantum state and the actual state produced by the control pulses. A potential disadvantage of these methods is that they may not always produce optimal results, depending on the specific properties of the quantum system being

optimized.

Chopped RAndom Basis (CRAB)

The CRAB protocol [17] works by randomly chopping the control pulses into a series of smaller segments, and then optimizing each segment independently using a gradient-based algorithm.

One of the main advantages of the CRAB algorithm is that it can achieve faster convergence and better optimization results than other techniques. However, this has a critic downside, due to the independent optimization of the chopped parts the resulting pulse may be discontinuous. In the worst case that means that the pulses generated cannot be implemented in real hardware. But in the average case, due to the discontinuities, the pulses generated have very high frequencies that may cause the quantum system to jump to other states outside of the computational basis.

GRadient Ascent Pulse Engineering (GRAPE)

The GRAPE protocol [18] works by iteratively updating the control pulses using a gradient ascent algorithm, in order to maximize the desired quantum gate operation or state transfer.

Compared to the CRAB algorithm, one of the main advantages of GRAPE is its ability to optimize the entire control pulse sequence at once, rather than optimizing each small segment individually. This generates smoother and continuous pulses with lower frequencies, that are easily implementable in real hardware. However, this might not be the case for every run, as it can be seen in figure 2.1. This is because gradient-based protocols are very sensible to the seed pulses with which they are initialized. One of the main drawbacks of GRAPE is that it fails to converge in a reasonable time as the system grows, which is a clear disadvantage.

1.4 Objectives

As we have seen in 1.1.2, an appropriate way to efficiently scale current quantum computing systems and be closer to error corrected quantum computers is to make use of modulable quantum subsystems, and interconnect them. In order to do so we need robust long-range quantum information transfer protocols. State of the Art (SOTA) protocols are either too slow, do not attain sufficient fidelities, are unfeasible in reality or do not take into account hardware limitations and/or external noises. In this work we aim to design and develop a digital system based on RL techniques that addresses all those needs and constraints at once.

High Velocity and Fidelity

In order to successfully transfer quantum information, as we are dealing with probabilities, we need to attain the maximum fidelity possible. This means that at the end of the protocol as much population of the system as possible is in the target state, to be able to confidently state that most of our information has been successfully transferred. There already exist protocols that reach fidelities of 100%, such as STA, so one may think that this issue has already been solved. The problem with these protocols is that they reach such fidelities only in ideal quantum systems. These kind of systems are not reproducible in reality, due to decoherence effects that appear when the quantum system interacts with the environment, and the existence of noise and decay. This is the reason why we need to perform quantum information transfer as fast as we can and, if possible, visiting as few intermediate states as possible, to prevent the population from interacting with those states and potentially be lost in the process.

Noisy Training

Current approaches to quantum information transfer protocols tend to work with ideal quantum systems and/or ideal pulse generators. That is not the case in reality, the reason is that the hardware equipment that interacts and controls the quantum system has a finite precision, meaning that is going to be imperfect and noisy. The digital system designed and developed in this work should take that into consideration and handle it appropriately, reducing the impact of hardware limitations and taking into account the existence of noise.

2

Design and Development

2.1 Recreate State of the Art Protocols

The first milestone we have to achieve to be able to design new quantum communication protocols is to explore the already existing ones. Appreciate their weak-points and develop solutions to mitigate them, comparing these SOTA protocols with our new protocols. In this case, the CTAP, STA, CRAB and GRAPE protocols were implemented, recreating all the figures that are shown on their respective papers. A *Pulses* script with pulses generators for the different algorithms stated above was developed for that purpose.

In order to verify that the implementations are valid, apart from comparing the pulses shapes and energies, we must apply them on a quantum system. To do so, one has to either have access to a real quantum system or simulated one. In this work we will use of the later. For that purpose we used the *QuTiP* library [23]. This library makes it easy to simulate the evolution, dynamics and interactions of quantum systems, solving the pertinent differential equations. To simplify things further, the *QTransferLib* script has been developed and builds on top of *QuTiP* functions to abstract the problem from a general quantum system to a quantum transfer problem.

Once the *Pulses* and *QTransferLib* scripts have been designed and developed, we are ready to simulate the behavior of the different protocols, and see how they change their ideal behavior when noise is added to them. In figure 2.1 we can see the pulses shapes, populations evolution, and the effect of noise on the different protocols. We can appreciate that, as mentioned in section 1.3, CTAP is much slower than STA to achieve a total transfer of the population. On ideal conditions, most of them achieve a fidelity close to 100%, but we can see that some of them are more robust to noise than others. For example, CTAP seems to maintain its average fidelity as noise is applied to it, while STA, CRAB and GRAPE decrease their average fidelity by around 15% on the worst noise scenario.

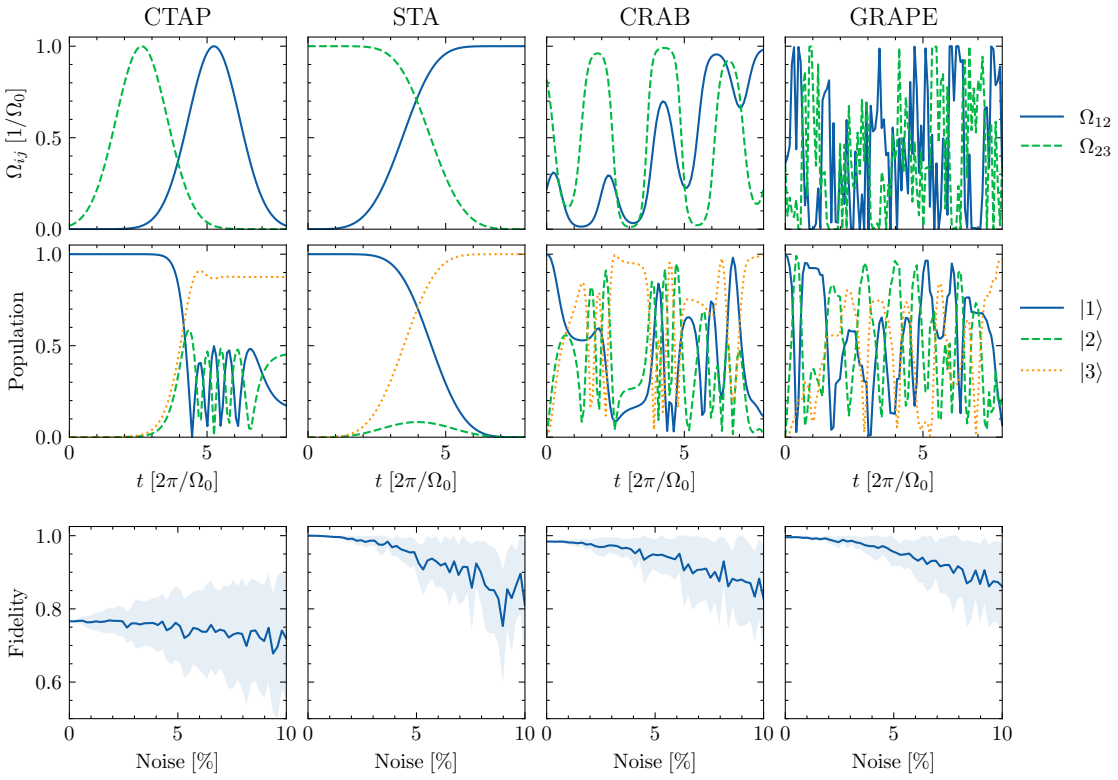

Figure 2.1: State of the art protocols and noise effects on them, one in each column: CTAP (first), STA (second), CRAB (third) and GRAPE (fourth). First row represents the protocol's pulses, for all the protocols $t_{max} = 7.96 * 2\pi/\Omega_0$. CTAP pulses were derived with $\sigma = t_{max}/6$, while STA pulses with $\alpha_0 = \Omega_0$, CRAB and GRAPE initial pulses were sampled from an uniform random distribution $U\{0, \Omega_0\}$. Second row shows the dynamics that these protocols induce on the populations of a three-level system initialized at state $|1\rangle$, with $\Delta_i = 0$. Third row represents the noise effects on each of the protocols, mean fidelity (solid) and standard deviation (shadow) are shown. The White Gaussian Noise (WGN) was added to the protocol pulses.

2.2 Suitable Architectures for the Problem

The main challenge of the problem we are tackling is that the possible combinations of pulses to drive the Hamiltonian, exponentially grows as the system gets bigger, and thus its search space cannot be fully explored in a reasonable amount of time. Taking that into account, a suitable architecture would be one based on Deep Reinforcement Learning. These type of algorithms are perfect to find a local or optimal minimum in a multidimensional space, making educated guesses based on the outcome of the actions that they apply to their environment, which in this case is the quantum system.

Apart from that, the pulses generated by our algorithm should be able to take continuous values. Deep reinforcement agents can be classified as discrete, like Deep Q-Network (DQN) [24], Proximal Policy Optimization (PPO) [25] and continuous like Twin Delayed DDPG (TD3) [26] and Soft Actor-Critic (SAC) [27]. This means that they search on a discrete or continuous actions spaces respectively. In our case we would like to make use of a continuous action space, since that is one of the properties of the hyperspace we are working with (pulses can take any continuous value). On top of that, our architecture should check the state of the quantum system as little as possible. Ideally it should be able to work just with the initial state of the system, the desired target state and the available time for the information transfer. In our case, rather than blindly surfing the vast hyperspace of possible pulses, we will make educated guesses by taking advantage of the fact that we are working on a simulation, and that we can get precise measurements of the state of our quantum system at any moment from it. All of this let us to choose the SAC algorithm, which is a deep reinforcement learning continuous agent, with a higher sample efficiency than TD3.

SOTA algorithms work flawlessly on ideal systems, which cannot be reproduced on a real setup, due to noise affecting different parts of both the quantum and control/measurement systems. To tackle imperfect quantum systems, we will have to simulate the population leakage to the environment. On the other hand, to make the generated pulses as versatile and robust to noise as possible, we need to model noise effects in the quantum system as well as in the control and measurement electronics. To make the model as general as possible we will use a well known technique used in robotic systems. In robotics, to transfer the behavior of the robots from simulation to real imperfect hardware, noise is added during the training on the simulation [28]. For our purpose we will artificially inject noise during the training process of our agent, in the actions (pulses) generated by it. This will help to generate pulses that are more robust to noise than SOTA ones.

2.3 Architecture and Infrastructure

As stated in the previous section, we have developed upon an architecture based in deep reinforcement learning, which must be as flexible and monitorizable as possible, to rapidly iterate through experiments. In subsection 1.2.1 we introduced the different modules that generally form a deep reinforcement system. In this section we will discuss the specific details of their design and implementation as well as the infrastructure needed to train and monitor the whole system. All the code developed in this section can be found at https://github.com/pabolojo/TFG-MLQITransfer

2.3.1 Architecture

To not over-engineer our system and achieve a desirable solution in a reasonable time, we have decided to use as our ground base library *TF Agents*. This is a module of the famous open-source project *TensorFlow* developed and maintained by Google. This frameworks makes it easy to experiment with SOTA architectures, being confident that the underlying implementation is correct and versatile to be implemented and optimized on different use cases and hardwares.

As we discussed in the previous section, we have chosen to use a SAC agent, this agent is already preimplemented in the *TF Agents* framework. The agent comes as a standalone module and we need to set the appropriate components for it to interact with, such as the environment, the cost functions and the replay buffer. Those components should be easily customized by tuning their hyperparameters. The overall system schematic architecture is shown in figure 2.2.

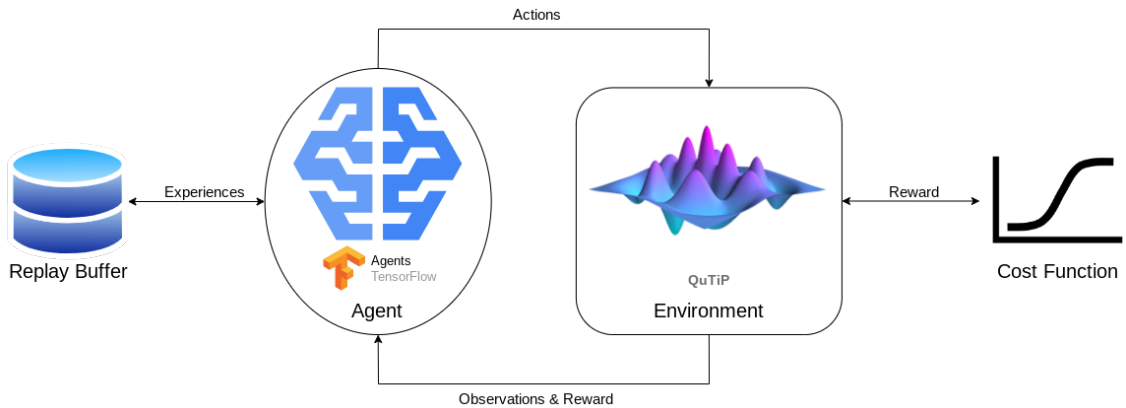

Figure 2.2: Deep Reinforcement Learning Architecture. Replay Buffer: stores the agent's past experiences with the environment and uniformly samples them when requested by the agent. Agent: generates the actions to interact with the environment based on previous experiences and their associated reward. Environment: simulates the effects of the actions generated by the agent on the quantum system. Cost Function: continuous reward given to the agent related to the distance between the current state and the target state the agent should reach.

Agent

This agent has a vast amount of hyperparameters that could be tuned, but the hyperparameters that are given on the original paper [27] have shown to work fine for a wide variety of scenarios. That is why we are only going to discuss some of them, the ones that we have seen that affect the most to our particular problem.

The SAC agent is formed by two neural networks, namely the actor and critic networks, whose functions were explained in subsection 1.2.1. Each of them consists of a different amount of layers and neurons and have their own corresponding learning rate, that can be fine tuned for better performance.

SAC works by entropy regularization, meaning that it adapts its exploration versus exploitation rate dynamically. The entropy regularization term encourages exploration in the policy by discouraging overly deterministic actions, promoting policies that have higher entropy or randomness. To control this ratio we have two hyperparameters, the α-*learning rate* and *target entropy*. The first one controls the alpha loss, which is used to update the *"temperature"* parameter, which controls the trade-off between the expected reward and the entropy of the policy distribution. The target entropy is used to guide the adjustment of alpha in the entropy regularization term.

And last but not least, another important hyperparameter is $gamma$, which is a discount factor for future rewards. Meaning that it is a parameter that determines the importance of future rewards compared to immediate rewards. It quantifies how much the agent values immediate rewards versus rewards obtained in the future.

Environment

Environments are a fundamental part of the architecture, in our case, the environments will be based on the quantum system simulation library *QuTiP*. To allow the *TF Agent* to interact with them, they have to implement a set of methods inherited from the *PyEnvironment* class, that act as an interface. Several environments had to be created to let the agent interact with different scenarios. All the environments have been created to be able to accept a variable number of states, different protocol duration times, variable number of steps, different initial states and target states, maximum (Ω_0) and minimum amplitudes of the generated pulses. The user can also define the energies Δ_i of the control Hamiltonian (equation (1.4)). Other than that, to include noise in the pulses, a $noise$ object from the *Noise* script can be passed to the environment. This noise is added to the original pulses following equation:

$$\Omega_{ij}(t)^{\text{noisy}} = \Omega_{ij}(t) + \eta(t) \ , \tag{2.1}$$

where $\eta(t)$ is a random value, sampled from a normal distribution, centered at 0, with standard distribution related to the noise percentage we want to model.

Before continuing this section it is worth making the difference between environment and *external*

environment, the former refers to the RL environment, while the latter refers to the *external environment* of the quantum system itself. As mentioned before, a set of environments have been created to support ideal systems (*QEnvWave*), systems with collapse operators (*QEnvDM*) and systems that interact with their *external environment* (*QEnvDecay*). There are mainly two types of environments, the ideal ones, where its wave-function evolves following the Schrödinger equation and thus at the end of our simulation we also get a wave function. And the ones that involve density matrices due to the implementation of collapse operators or interactions with the *external environment*. For this type of environments, collapse operators can be easily passed to them when initialized.

The environments have a maximum duration, but the agent is not forced to spend all those time steps. This means that each episodes can have different duration, rewarding the agent if it completes the task in fewer steps, performing the transfer in a sorter time. Furthermore, to evaluate the agent after being trained and to be able to retrieve the best pulses that led to the maximum reward, we periodically save the best episodes pulses, populations and cost functions. This is very useful for future analysis on experiments and allows to quickly iterate on them by tweaking the hyperparameters of the system.

Cost Functions and Custom Metrics

Cost functions are one of the key points for an agent to really be able to learn and make educated guesses based on its previous experiences. It is responsible for guiding the agent on its way to find the optimal solution for our problem. In our case we have started by stating the three base scenarios that the agent may find:

- No Transfer: This is the worst scenario, and is one the agent is likely to fall in if it is punished excessively for other factors such as populating intermediate states. Ideally we would prefer that any transfer happens, even if it implies populating not desired states in the process, rather than the system staying in the same initial state.
- Sequential Transfer: Having successfully achieved information transfer between the initial and target state, this is our second worst scenario. We have achieved that transfer but we have populated maximally the intermediate states. We already know that this solution would work. But in real systems with noise we would like our population to populate minimally the intermediate states, to reduce the amount of interactions with the *external environment*.
- Optimal Transfer: We know that for a three-level ideal system, STA is the theoretically derived optimum protocol. We should aim for the agent to achieve the kind of pulses that STA generates on a three-level system. These pulses allow for the fastest information transfer possible, while populating minimally the intermediate state.

Our hypothesis is that the desired cost function that we are looking for should be a continuous linear combination of the populations, that returns a increasingly higher reward depending on the scenarios explained above (from worst to best). The premises from which we have constructed and ordered those scenarios are:

- Some information transfer is better than no information transfer at all (*Reward*(No Transfer) < *Reward*(Sequential Transfer)).
- Not populating intermediate states is better than populating them, if the same final fidelity is achieved on an ideal system (*Reward*(Sequential Transfer) < *Reward*(Optimal Transfer)).

- Preserve the population on the target state, rewarding faster information transfers and punishing population oscillations between states.

Our first approach to build this desired cost function was with a linear combination of the populations of each of the states, being the reward a weighted sum of them. Applying proportional weights to each of the state populations. We can see in figure 2.3 (*P*) that we manage to model a cost function that satisfies most of our premises, the only problem is that the final reward achieved by the *Sequential Transfer* and *Optimal Transfer* scenarios is the same, but we obviously want to aim for the later.

This can be achieved by punishing the agent in the long run, so that it remembers it can improve. There are two ways of punishing the agent: directly in the cost function or by adding decay terms on the environment, so that the simulation also imposes punishments. In case of the cost function, this is implemented by introducing integral weights, that act as a weighted sum of the sums of each of the populations until that moment. As we can see in figure 2.3 (*PI*) *Sequential Transfer* and *Optimal Transfer* do not achieve the same final reward now.

With this cost function we wont be preventing oscillations that could happen between non punished states (i.e. if the population start oscillating from state $|1\rangle$ and $|3\rangle$) without populating $|2\rangle$). We would also like to reward the fastest protocol, meaning the one that populates the target state faster. For that we introduce some derivative weights that act on the difference between current and past populations. The achieved final cost function can be seen in figure 2.3 (*PID*) and is defined by equation:

$$\mathcal{C}(t) = \sum_{i}^{N=3} \left(P_i \rho_{ii}(t) + I_i \int_0^t \rho_{ii}(t')dt' + D_i \frac{\partial \rho_{ii}(t')}{\partial t'}\bigg|_{t'=t} \right) . \tag{2.2}$$

Note that, when working with pure states $\rho_{ii}(t) \rightarrow |\langle i|\psi(t)\rangle|^2$ in the equation above. For some of the readers this might remember to the well known approach taken in control theory named Proportional Integral Derivative (PID) .

Other important parameter of the cost function is the *end episode threshold*, which allows to set a reward threshold that, if achieved, will directly end that episode. This allows to have variable length episodes and added to the discount factor, it motivates the agent to find shorter protocols.

Regarding the custom metrics, we have added the *intermediate population* and *target population* metrics. The former logs the maximum of the intermediate populations during each of the episodes, while the latter logs the final expectation of the system at the end of each episode. Apart from these custom metrics, a set of standard metrics such as the average return of the agent, the loss of each of the networks and the length of each of the episodes are also logged to be able to monitor the experiment while the system is being trained and also to analyze experiments when they have finished.

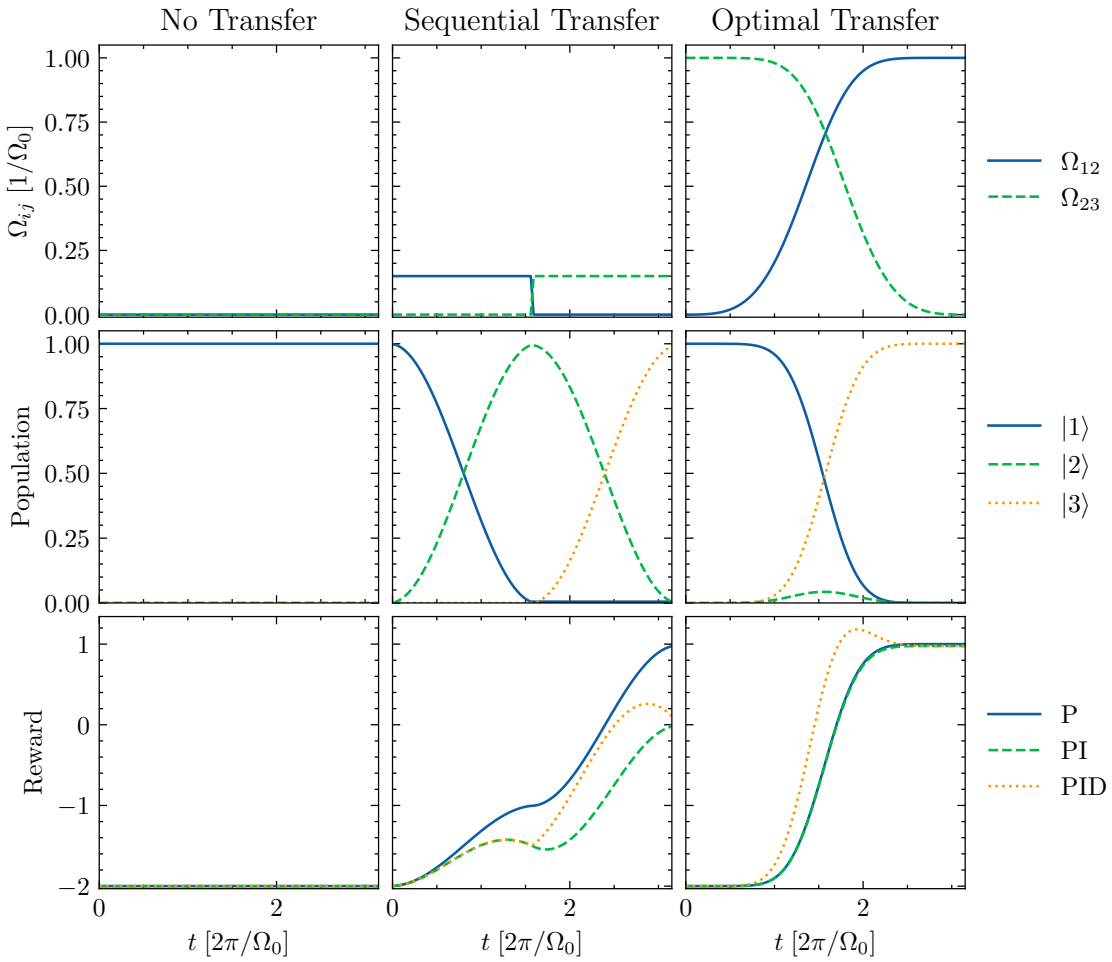

Figure 2.3: Cost functions comparison between three base scenarios, one in each column: no information transfer (left), sequential information transfer (center) and optimal information transfer (right). This last scenario has been obtained by the STA protocol with $\alpha_0 = \Omega_0$ and $t_{max} = 3.18 * 2\pi/\Omega_0$. First row represents the protocol's pulses for each of the scenarios. Second row shows the dynamics that these protocols induce on the populations of a three-level system initialized at state $|1\rangle$, with $\Delta_i = 0$. Third row represents the reward returned by cost functions with P (blue, solid), PI (green, dashed) and PID (orange, dotted) weights, associated to the evolution of each of the quantum systems. The parameters are P = {-2, -1, 1}, I = {0, -0.03, 0}, D = {0, 0, 25}.

Replay Buffer

A replay buffer should be accessible by the agent so that it can remember what happened in its previous interactions with the environment. This reduces the amount of interactions of the agent with the environment and thus the amount of measurements too, being more sample efficient. In our case we are using a *TFUniformReplayBuffer* which is uniformly sampled by the agent. The buffer can be configured with the following parameters:

- Replay buffer capacity: is the capacity of the replay buffer, it should be a multiple of the number of steps of the environment.
- Initial collect steps: is the amount of step interactions with the environment that are added to the replay buffer before starting the training. It should be lower than the overall capacity of the buffer.

Before starting the training, the buffer is prefilled with *initial collect steps* interactions with the environment, so that the agent has a partially filled buffer to interact since the beginning of its training.

2.3.2 Infrastructure

Once our architecture has been designed and developed it is time to test it, for that we need an appropriate infrastructure that is capable of handling the huge amount of computations required for the agent to be trained, the replay buffer to be filled and the environment to be simulated. For that we need a distributed network of High Performance Computing (HPC) , commonly known as an HPC cluster. We also need to be able to monitor the system while it is training and after experiments have finished, this is achieved by the use of metrics.

HPC Cluster

To train the system we have used the Drago Cluster provided by the Consejo Superior de Investigaciones Científicas (CSIC) . This cluster has access to the latest Graphics Processing Units (GPUs) and Central Processing Units (CPUs) , reducing the training time several orders of magnitudes from the one it will take in a consumer computer. Having implemented the architecture using the *TensorFlow* library is a huge advantage, as it natively comes with parallel computation in mind. Making our architecture suitable to be executes in a distribute computing network out of the box. In fact, *TensorFlow* code is really optimised to be run on GPUs.

Other than that, the replay buffer of the architecture should be kept in Video Random Access Memory (VRAM) memory to avoid cache misses and loadings from main memory that would increase the overhead needed to access it. When filled, the replay buffer size is of several gigabytes. Apart from that, the neural networks as well as the simulation variables and states should also be loaded in VRAM for the same reason as the replay buffer. All of this makes it almost impossible for our architecture to be run on a conventional consumer system, that usually counts with 8-12 gigabytes of VRAM.

All of this makes the Drago Cluster suitable to run and train our system, which is trained on Nvidia A100 GPUs that have 40 gigabytes of VRAM, enough to feet all the necessary components of our system. The cluster is managed by Simple Linux Utility for Resource Management (SLURM) , a software that manages all the infrastructure of the cluster and lets you run and schedule jobs.

Monitor Metrics

Other aspect that should be taken into account while training machine learning models is monitorization. This consist of generating different system metrics such as network errors or agent rewards. This is done to make sure the agent is learning in the appropiate way, being able to stop the current experiment in case the desired results are not being achieved. After an experiment is run, metrics allow to quickly iterate on the tuning of the system hyperparameters between experiments.

TensorFlow allows to set up metrics loggins on commonly used information such as network losses, agent average returns per episode or average episode lengths. As well as allowing developers to add their own metrics. In our case we are interested in monitoring the neural network losses, the average return of the agent, the episode lengths and the custom metrics we discussed in the previous subsection.

To allow the appropriate visualization of that logged information, we setup a TensorBoard server. This server is a web interface that automatically parses the logged metrics information and presents it to the user in a convenient way, like in plots or charts. With this setup, one can compare experiments, check current experiments status and make educated guesses on how to tune the hyperparameters of the system for future experiments.

3

Experiments and Results

Once the whole system has been designed, developed and the infrastructure needed to execute it was configured and available, it was time to start our rounds of experiments. Our first goal was to demonstrate that our system was indeed suitable for generating pulses that achieve fast and high fidelity quantum information transfer. After achieving that milestone, we wanted to check if the pulses generated by our algorithm were more robust to noise than SOTA ones. And finally, we have explored the effects of the pulses that our system generates for larger quantum systems, were no SOTA referent exist (apart from the extension via straddling pulses of the already know SOTA protocols).

3.1 Finding STA on three-level systems

As we have mentioned in subsection 1.3.2, it can be theoretically proven that STA is the fastest alternative to CTAP, achieving the same efficiency in a three-level ideal quantum system. Therefore we can confidently state that STA is the fastest protocol that achieves the highest fidelity on a three-level ideal quantum system.

Our first goal would be to demonstrate that our RL-based system can produce high fidelity and fast protocols. We could directly train it on an arbitrarily long chain of qubits, but that would be computationally expensive and it would take to much time to fine tune the system's hyperparameters, since we would not have any reference pulses to compare our results with. This is the reason why our first approach to prove our system valid and be able to fine tune the hyperparameters of all the modules has been to train it on the simplest quantum system possible. Of course it would not make sense to train it on a one-level quantum system, since all the information is already localized on that one same level. For the case of a quantum system with two levels, we could argue that if our system finds how to perform an optimal sequential transfer (which it does) via a π-*pulse* we could say that the system is able to generate the desired pulses. This is true in some sense, but to really be able to claim that, a more interesting quantum system to apply our algorithm to is an ideal quantum system (no interactions with the *external environment*) with three-levels and with no noise applied to the pulses during the system's training. Our goal then would be for our system to produce STA-like pulses, which means achieving a

similar fidelity than the one STA achieves on the same time or less, by also maintaining the intermediate state with as low population as possible.

Our first attempt was with the simplest cost function we came up with, the reward was just the population on the target state. With this approach without further constrains, after a few hours of training, the system found a protocol that achieves the highest fidelity in the shortest time possible, but without taking into consideration the intermediate population. The protocol that the agent found was simply two π-*pulses* applied each to one of the coupling gates. This allows the quantum system to perform a quasi-sequential transfer between $|1\rangle$ and $|2\rangle$ and then from $|2\rangle$ to $|3\rangle$, while highly populating the intermediate state in the process. At this stage we could not blame the agent for not finding STA-like protocols, since it was doing what it was meant to do with the constraints it was given.

Seeing those results, we arrive to the conclusion that our cost function should be more complex, that is why we trained our system with a *P* cost function like the one showed in 2.3. This cost function allows the agent to associate populating $|3\rangle$ with a much higher reward than populating $|2\rangle$. With this new cost function we managed to reduce the intermediate population considerably, as shown in figure 3.1, but we would like to lower it more.

The best results were achieved when a *PID* cost function, like the one showed in 2.3, was given as reward to the agent. As it can be seen in figure 3.1, with this cost function the agent was able to find STA-shaped pulses that reached a fidelity of 97%, while populating the intermediate state 10%. Compared to STA, which achieves a fidelity of 100% and an intermediate population of 5% in a fraction of time more, there is yet a bit of room for improvement, but the results were quite impressive. We cannot forget that this was achieved without any a-priori, the agent figured out by itself a protocol that satisfies the conditions imposed and arrived to the conclusion that the best pulses are counter-intuitive, as we discussed in paragraph 1.1.3.

Given that we did not want to over-complicate the cost function in excess, our next approach was to implement the punishment for populating the intermediate state directly into the simulation, by simulating population leakage to the *external environment* if the intermediate state is populated. This is done in the *QEnvDecay* environment, where an extra state is added to the system. This extra state is only coupled with the intermediate state and thus the leakage to the *external environment* is simulated by transferring population to that extra state. The leakage rate was set such that, if applied to STA, it will attain a fidelity of 80%. With this new environment and the simplest cost function (population $|3\rangle$ of the system *P* = {0, 0, 1}), our system achieved similar results than with a *PID* cost function, as seen in figure 3.1. Note that for the results shown in the figure, no leakage is included, this is only present during the training of the system.

We can see in table 3.1 that all the pulses obtained by the RL algorithm have lower energy than STA pulses. They also perform the information transfer in less time or equal compared to STA. The trade-off for being more energy efficient and faster is that they achieve lower fidelities and populate the

intermediate state more compared to STA.

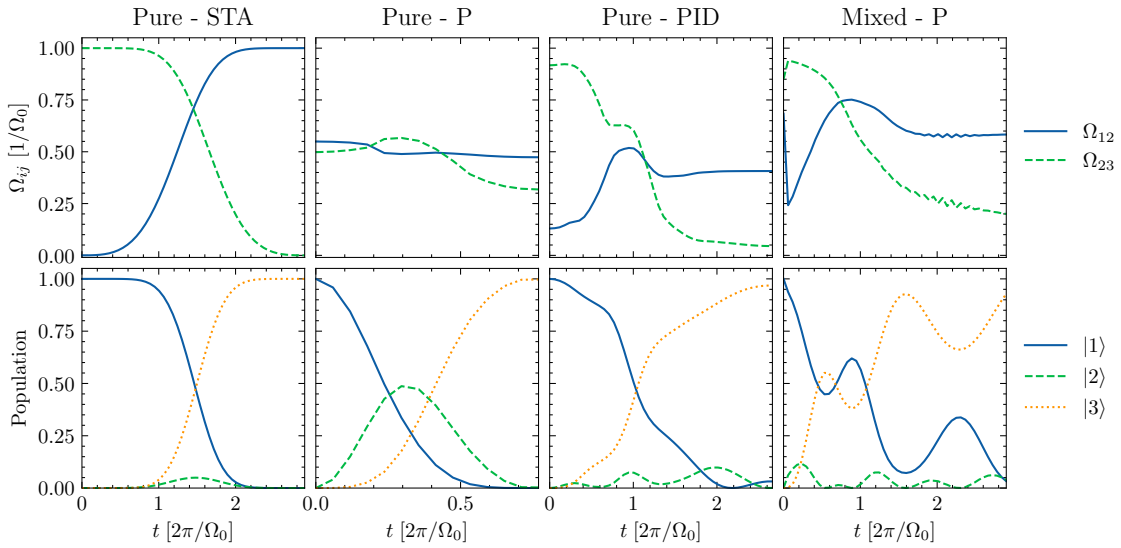

Figure 3.1: Agent results on three-level pure systems for different training environment types and cost functions. Each column represents a different scenario: STA protocol applied to pure system (first), RL rewarded with P cost function trained on pure system environment (second), RL rewarded with PID cost function trained on pure system environment (third) and RL rewarded with P cost function trained on mixed environment (fourth). The first scenario has been obtained by the STA protocol with $\alpha_0 = \Omega_0$ and $t_{max} = 2.90 * 2\pi/\Omega_0$. First row represents the protocol's pulses for each of the scenarios. Second row shows the dynamics that these protocols induce on the populations of a three-level pure system initialized at state $|1\rangle$, with $\Delta_i = 0$.

	Pure - STA	**Pure - P**	**Pure - PID**	**Mixed - P**
Energy [Ω_0]	2.90	0.52	1.61	2.33
Max. I. P.	0.05	0.49	0.10	0.12
Fidelity	1.00	1.00	0.97	0.93
t_{max} **[$2\pi/\Omega_0$]**	2.90	0.77	2.66	2.90

Table 3.1: DRL and STA protocols properties on three-level pure systems. Each row is one property: Energy (first), Maximum Intermediate Population (second), Fidelity (third), Total transfer time (fourth).

Environments with decay are computationally more expensive compared to the pure ideal ones, since they involves density matrices and have an extra state to handle, to simulate the leakage of population to the *external environment*. But they only relay on the simplest cost function (population $|3\rangle$ of the system $P = \{0, 0, 1\}$) and the decay rate for the intermediate states, which is fixed depending on its effects on STA. On the other hand, simple ideal environments (*QEnvWave*) that are trained with a *PID* cost function, are faster but have a greater amount of parameters to be fine tuned (extra *PID* weights). In view of having achieved similar results with both approaches, for our next phase of experiments we decided to stick with the decay environment (*QEnvDecay*), since it has only $N - 2$ parameters to optimize compared to the $3N$ parameters that would require an N states system.

3.2 Noise effect on DRL protocols vs SOTA protocols

As we mentioned in section 1.4, we would like to see if our system can generalize well and generate pulses that are robust to noise, or at least more robust than SOTA protocols. We have seen in figure 2.1 that with more than 3% of noise, SOTA protocols start to dramatically reduce their fidelity, and at around 10% of noise, most of them have decreased their mean fidelity up to 20%, as it is the case for STA.

With all of that in mind, our next round of experiments was focused in generating new protocols with our RL architecture that were robust to noise. For that, the system was trained with the same setup and hyperparameters obtained in the previous round of experiments, but this time with 5% White Gaussian Noise (WGN) . This noise was artificially injected into the actions (pulses) that the agent generated. Doing so, the agent is not aware at any moment that those actions are being randomly modified due to the WGN that is being added to them, and then they are applied to the environment (quantum system). In this way we force the agent to generate pulses that may be subject to random modifications, and thus making the generated pulses more robust to noise.

Once our system has been *noisy-trained*, it was time to compare the generated protocol with SOTA ones. To make the comparison fair, the RL algorithm was trained on episodes of the same length than the ones of CTAP, STA, CRAB and GRAPE. The architecture was trained on a 5% noise intensity, and then sampled the noise-free generated pulses with different noise intensities. One could also train several models with different noise intensities and may certainly achieve better results for each of the individual noise discretizations. The results of our experiment are shown in figure 3.2, we can see that, overall, our RL noisy generated protocol is more robust to noise compared to SOTA protocols, achieving higher fidelities than SOTA protocols as we increase the noise intensity. In table 3.2 we can see the fidelities achieved by each of the algorithm when subjected to the maximum noise (10%), we notice that our protocol has a the higher fidelity (98%) and the lower standard deviation (2%). This confirms the hypothesis we proposed in section 2.2, training the agent in a noisy environment has forced it to generate protocols that are robust to WGN.

	CTAP	**STA**	**CRAB**	**GRAPE**	**DRL**
Fidelity (10% noise)	0.72 ± 0.17	0.81 ± 0.17	0.83 ± 0.15	0.86 ± 0.12	0.98 ± 0.02

Table 3.2: DRL vs SOTA protocols fidelities under maximum noise effects (10%). Mean $\pm$ standard deviation are shown for each of the protocols.

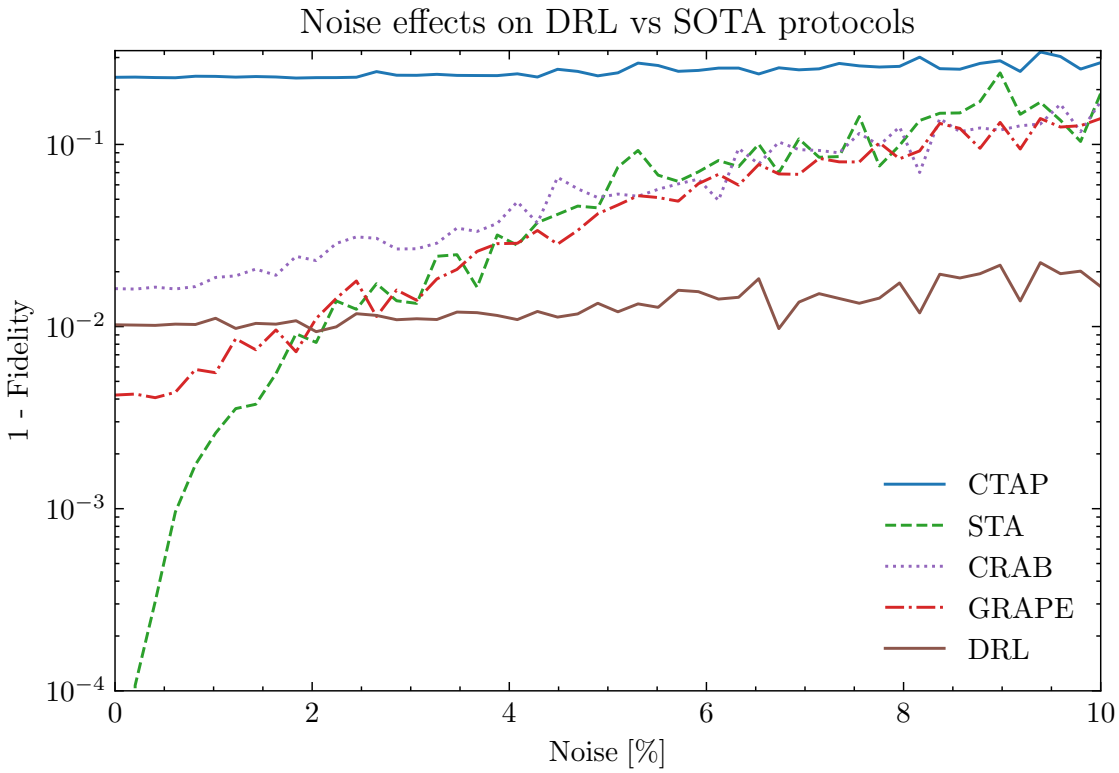

Figure 3.2: Noise effects on SOTA protocols and the DRL generated protocol. CTAP mean infidelity (solid, blue), STA mean infidelity (dashed, green), CRAB mean infidelity (dotted, purple), GRAPE mean infidelity (dashdot, red), DRL mean infidelity (solid, brown). With $t_{max} = 7.96 * 2\pi/\Omega_0$ for all the protocols except for DRL, where $t_{max} = 1.62 * 2\pi/\Omega_0$. The CTAP protocol has been obtained with $\sigma = t_{max}/6$ and STA with $\alpha_0 = \Omega_0$. The infidelity is represented in a logarithmic scale.

3.3 Beyond three-level systems

We have discussed that SOTA protocols such as CTAP and STA, were originally designed to work on three-level systems. Despite that, there exists a technique to generalize them to quantum dots chains of arbitrary length. The downside of this technique is that it only works on chains that have an odd number of states, since it can be proven that they are the only ones where a dark state exists. If the protocol used to drive the system is compliant with the adiabaticity theorem, the evolution of the system can be guided by the dark state. The technique to expand the protocols to odd systems larger than three-level is based on adding *straddling* pulses to the couplings of the intermediate states. These pulses span the energy gap between adjacent intermediate states, making them more difficult to populate. The higher the energy of those straddling pulses, the lower the populations in the intermediate states.

All of this has a trade-off, as we increase the energy of the straddling pulses we might be getting lower populations on the intermediate states in an ideal quantum system. But that is not the case on a real mixed quantum system, where more energy applied to the systems means that it is more likely to transition to unwanted states outside of our base states (like when an electron absorbs a photon and jumps to the next orbital). This is why we would like to find protocols that are energy efficient (i.e. they have low amplitude smooth pulses) and that maintain the intermediate populations as low as possible.

In our simulations, we have challenged our algorithm to find better protocols than the one generated by combining STA and the straddling pulses technique for a five-level system. We have also focused our attention in the pulses that our RL algorithm could find in a four-level system. In this case we cannot compare the generated protocols with the ones derived by the straddling technique, since it only works on systems with an odd number of states. For this case we compare it to the protocol formed by sequentially applying STA and π-pulses, in that way the transfer would happen on two stages, first from $|1\rangle$ to $|3\rangle$ and then sequentially from $|3\rangle$ to $|4\rangle$. We have decided to study four and five-level systems because they are the less computationally expensive systems above three-level systems, and would be enough to confirm that our technique can be extended to any number of states, given the sufficient computational resources and time for the algorithm to be trained are available.

As we did for the experiments conducted in section 3.2, for these experiment we are using a *QEnvDecay* environment with a simple population of $|3\rangle$ (P = {0, 0, 1}) cost function. The decay rates have been set equal for all the intermediate states. We have observed that in the case of five-level systems, applying the same decay rate to all the intermediate states generates hybridized states. These states are effectively decoupled from the environment, not being influenced by it. To avoid this effect as much as possible we have chosen a decay rate that achieved the lowest fidelity in STA, and therefore we minimize the impact of the hybridized states on the training. The Ω_0 parameter has been set as the maximum amplitude of the straddling pulses. The parameters for the four-level experiment have been derived by interpolating the ones from the five-level system and the three-level system.

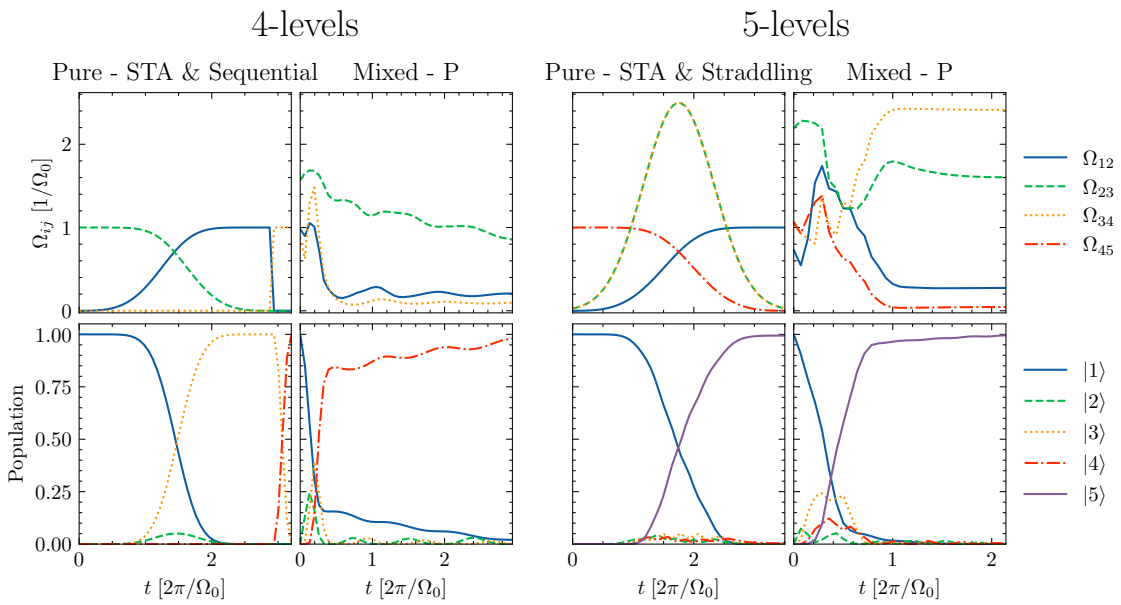

Figure 3.3: Agent results on four and five-level pure systems. Each column represents a different scenario: STA & Sequential protocols applied to four-level pure system (first), RL rewarded with P cost function trained on four-level mixed environment (second), STA & Straddling protocols applied to five-level pure system (third) and RL rewarded with P cost function trained on five-level mixed environment (fourth). The STA protocol part of the first and third scenarios has been obtained with $\alpha_0 = \Omega_0$ and $\Omega_{s0} = 2.5\Omega_0$, while $t_{max} = 2.90 * 2\pi/\Omega_0$ for the first one and $t_{max} = 3.50 * 2\pi/\Omega_0$ for the third. First row represents the protocol's pulses for each of the scenarios. Second row shows the dynamics that these protocols induce on the populations of a four-level pure system (first and second columns) and five-level pure system (third and fourth columns) initialized at state $|1\rangle$, with $\Delta_i = 0$.

	4-levels		**5-levels**	
	Pure - STA & Sequential	**Mixed - P**	**Pure - STA & Straddling**	**Mixed - P**
Energy [Ω_0]	3.20	3.67	6.72	6.06
Max. I. P.	1.00	0.39	0.05	0.24
Fidelity	0.99	0.98	0.99	1.00
t_{max} **[$2\pi/\Omega_0$]**	3.20	2.94	3.50	2.14

Table 3.3: DRL and STA protocols properties on four and five-level pure systems. Each row is one property: Energy (first), Maximum Intermediate Population (second), Fidelity (third), Total transfer time (fourth).

The results of these experiments are exposed in figure 3.3. We can see that, for a four-level system, our algorithm has found a better protocol than the one produced by concatenating STA and π-pulses. It can be seen in table 3.3 that it achieves a comparable fidelity in a shorter amount of time. The main advantage of the protocol generated by the RL algorithm for this scenario of a four-level system is that the intermediate population is way lower than the one we obtain by the SOTA approach.

Regarding the five-level experiment, as it can be seen in table 3.3 we even attain better fidelities in a shorter time than with the SOTA approach of extending STA with straddling pulses for five-level systems, although we populate the intermediate states more. Furthermore, the pulses generated with RL are more energy efficient than the SOTA ones, making them less likely to transfer population into unwanted states outside of the system.

In view of these results, we can confidentially state that our RL-based architecture is extensible to an arbitrary number of states, and is capable of finding protocols with comparable quality than SOTA ones or in some cases even better.

4

Conclusions and Future Work

4.1 Conclusions

In this work we have demonstrated that Deep Reinforcement Learning architectures are suitable for finding optimal quantum information transfer protocols. We know that STA is the analytically-derived optimal solution for a three-level system. As shown in section 3.1, our technique was able to find STA-like optimal protocols that share key characteristics with it such as being counter-intuitive, speed, fidelity and intermediate populations. Therefore, having found similar protocols with our new approach is a promising evidence to claim that this could also be the case for larger systems.

We have also proven that our system is capable of generating noise-robust protocols, as it was discussed in section 3.2. This opens the possibility of using the generated protocols in a wide variety of setups, and being likely for them to work properly. Having the property of being robust to noise, little variances between different setups may be well assimilated by the protocol.

And finally, we have showed that our architecture is extensible to larger systems, even achieving better results than SOTA protocols, as we saw in section 3.3. This confirms that our approach can be extended to an arbitrary system size, as long as the sufficient computational resources and time are available, it will generate suitable protocols for it.

In view of all these results, we can confidently state that our system is capable of being particularly trained for a specific system or setup and can generate noise-robust protocols for it. This would allow fine tuning the models for current real setups to scale them further by implementing the generated long-range quantum information transfer protocols between the smaller nodes of the aggregated system, potentially scaling SOTA quantum computers further.

4.2 Future Work

If this work wants to be continued, there are several research lines that could be followed. Future works could focus either in generating new protocols for the quantum systems tackled in this work, but using different techniques/hyperparameters. Also, finding new protocols for other arrangements of quantum systems.

For example, instead of having a per-step action, having a per-episode action. This would open the possibility to optimize ansatz parameterized functions such as the STA χ coefficients, generate points that define a Bezier continuous curve, or the coefficients of unitary transformations such as the Discrete Fourier Transform (DFT) , Discrete Cosine Transform (DCT) or Discrete Sine Transform (DST) .

Regarding different quantum system arrangements, another interesting research line is to study the protocols generated by our system for longer chains of quantum dots. The optimal protocol for each of the different chain lengths must be analyzed to spot patterns and create analytic formulas to generate optimal noise-robust protocols for chains longer than three quantum dots.

Last but not least, the approach taken in this work could be extended to 2D quantum systems (arrays of quantum dots instead of chains). This would allow the implementation of such protocols in semiconductor chips based on quantum dot arrays, and achieve on-chip communication between any pair of quantum dots. In this scenario, not being restricted to only one dimension, ideally we would like to perform several quantum information transfers simultaneously. This would involve much more sophisticated optimizations, since the different quantum information transfers may affect one another.

Bibliography

[1] L. Giannelli, P. Sgroi, J. Brown, G. S. Paraoanu, M. Paternostro, E. Paladino, and G. Falci, "A tutorial on optimal control and reinforcement learning methods for quantum technologies," *Physics Letters A*, vol. 434, p. 128054, 2022.

[2] 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 Information*, vol. 5, no. 1, 2019.

[3] R. Porotti, D. Tamascelli, M. Restelli, and E. Prati, "Coherent transport of quantum states by deep reinforcement learning," *Communications Physics*, vol. 2, no. 1, p. 61, 2019.

[4] V. Sivak, A. Eickbusch, H. Liu, B. Royer, I. Tsioutsios, and M. Devoret, "Model-free quantum control with reinforcement learning," *Physical Review X*, vol. 12, no. 1, p. 011059, 2022.

[5] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, ISBN 9781107002173, 2010.

[6] D. Manzano, "A short introduction to the lindblad master equation," *AIP Advances*, vol. 10, no. 2, p. 025106, 2020.

[7] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6, pp. 467–488, 1982.

[8] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997.

[9] S. Yoo, J. Bang, C. Lee, and J. Lee, "A quantum speedup in machine learning: finding an n-bit boolean function for a classification," *New Journal of Physics*, vol. 16, no. 10, p. 103014, 2014.

[10] N. S. Blunt, J. Camps, O. Crawford, R. Izsák, S. Leontica, A. Mirani, A. E. Moylett, S. A. Scivier, C. Sünderhauf, P. Schopf, J. M. Taylor, and N. Holzmann, "Perspective on the current state-of-the-art of quantum computing for drug discovery applications," *Journal of Chemical Theory and Computation*, vol. 18, no. 12, pp. 7001–7023, 2022.

[11] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven, "Characterizing quantum supremacy in near-term devices," *Nature Physics*, vol. 14, no. 6, pp. 595–600, 2018.

[12] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation," *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 439, no. 1907, pp. 553–558, 1992.

[13] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. van den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, and A. Kandala, "Evidence for the utility of quantum computing before fault tolerance," *Nature*, vol. 618, no. 7965, pp. 500–505, 2023.

[14] A. M. Tyryshkin, S. Tojo, J. J. L. Morton, H. Riemann, N. V. Abrosimov, P. Becker, H.-J. Pohl, T. Schenkel, M. L. W. Thewalt, K. M. Itoh, and S. A. Lyon, "Electron spin coherence exceeding seconds in high-purity silicon," *Nature Materials*, vol. 11, no. 2, pp. 143–147, 2012.

[15] A. D. Greentree, J. H. Cole, A. R. Hamilton, and L. C. L. Hollenberg, "Coherent electronic transfer in quantum dot systems using adiabatic passage," *Physical Review B*, vol. 70, no. 23, 2004.

[16] Y. Ban, X. Chen, and G. Platero, "Fast long-range charge transfer in quantum dot arrays," *Nanotechnology*, vol. 29, no. 50, p. 505201, 2018.

[17] P. Doria, T. Calarco, and S. Montangero, "Optimal control technique for many-body quantum dynamics," *Physical review letters*, vol. 106, p. 190501, 2011.

[18] B. Rowland and J. Jones, "Implementing quantum logic gates with gradient ascent pulse engineering: Principles and practicalities," *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 370, pp. 4636–50, 2012.

[19] A. J. Sigillito, M. J. Gullans, L. F. Edge, M. Borselli, and J. R. Petta, "Coherent transfer of quantum information in a silicon double quantum dot using resonant swap gates," *npj Quantum Information*, vol. 5, no. 1, p. 110, 2019.

[20] A. R. Mills, D. M. Zajac, M. J. Gullans, F. J. Schupp, T. M. Hazard, and J. R. Petta, "Shuttling a single charge across a one-dimensional array of silicon quantum dots," *Nature Communications*, vol. 10, no. 1, 2019.

[21] B. Rowland and J. Jones, "Implementing quantum logic gates with gradient ascent pulse engineering: Principles and practicalities," *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 370, pp. 4636–50, 2012.

[22] P. Doria, T. Calarco, and S. Montangero, "Optimal control technique for many-body quantum dynamics," *Physical Review Letters*, vol. 106, no. 19, 2011.

[23] J. Johansson, P. Nation, and F. Nori, "Qutip: An open-source python framework for the dynamics of open quantum systems," *Computer Physics Communications*, vol. 183, no. 8, pp. 1760–1772, 2012.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[26] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*, pp. 1587–1596, PMLR, 2018.

[27] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.

[28] T. Haarnoja, B. Moran, G. Lever, S. H. Huang, D. Tirumala, M. Wulfmeier, J. Humplik, S. Tunyasuvunakool, N. Y. Siegel, R. Hafner, *et al.*, "Learning agile soccer skills for a bipedal robot with deep reinforcement learning," *arXiv preprint arXiv:2304.13653*, 2023.

TERMINOLOGY

Aprendizaje por Refuerzo Profundo subcampo del aprendizaje automático que combina técnicas de aprendizaje profundo con principios de aprendizaje por refuerzo para permitir a los agentes aprender y tomar decisiones en entornos complejos.

cúbit unidad básica de información en computación cuántica (bit-cuántico).

Deep Reinforcement Learning subfield of machine learning that combines deep learning techniques with reinforcement learning principles to enable agents to learn and make decisions in complex environments.

Machine Learning branch of artificial intelligence and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

Noisy Intermediate Scale Quantum computers quantum computers that are prone to considerable error rates and limited in size by the number of logical qubits (or even physical qubits) in the system.

qubit basic unit of information in quantum computing (quantum-bit).

Acronyms

AI Artificial Intelligence.

CPUs Central Processing Units.

CRAB Chopped RAndom Basis.

CSIC Consejo Superior de Investigaciones Científicas.

CTAP Coherent Transport by Adiabatic Passage.

DCT Discrete Cosine Transform.

DFT Discrete Fourier Transform.

DQN Deep Q-Network.

DST Discrete Sine Transform.

GPUs Graphics Processing Units.

GRAPE GRadient Ascent Pulse Engineering.

HPC High Performance Computing.

ML Machine Learning.

NISQ Noisy Intermediate-Scale Quantum.

NN Neural Network.

PID Proportional Integral Derivative.

PPO Proximal Policy Optimization.

RL Reinforcement Learning.

SAC Soft Actor-Critic.

SLURM Simple Linux Utility for Resource Management.

SOTA State of the Art.

STA Shortcuts to Adiabaticity.

TD3 Twin Delayed DDPG.

VRAM Video Random Access Memory.

WGN White Gaussian Noise.

APPENDICES

A

Quantum Mechanics Postulates

Postulate 1: Associated to any isolated physical system is a complex vector, known as the state vector, defined with an inner product over a Hilbert space, known as the *state space* of the system. The system is completely described by its *state vector*, which is a unit vector in the system's state space.

Postulate 2: The evolution of a *closed* quantum system is described by a *unitary transformation*. That is, the state $|\psi\rangle(t_1)$ of the system at time t_1 is related to the state $|\psi\rangle(t_2)$ of the system at time t_2 by a unitary operator $U(t_1, t_2)$ which depends only on the initial and final times,

$$|\psi(t_2)\rangle = U(t_1, t_2)|\psi(t_1)\rangle \ .$$

Postulate 2': The time evolution of the state of a closed quantum system is described by the time-dependent *Schrödinger equation*,

$$i\hbar \frac{\mathrm{d}|\psi\rangle}{\mathrm{d}t} = H|\psi\rangle \ .$$

In this equation, $\hbar$ is a physical constant known as *Planck's constant*. In practice, it is common to absorb the factor $\hbar$ into H, effectively setting $\hbar = 1$. H is a fixed Hermitian operator known as the Hamiltonian of the closed system.

Postulate 3: Quantum measurements are described by a collection M_m of *measurement operators*. These are operators acting on the state space of the system being measured. The index m refers to the measurement outcomes that may occur in the experiment. If the state of the quantum system is $|\psi\rangle$ immediately before the measurement then the probability that result m occurs is given by

$$p(m) = \langle\psi|{M^\dagger}_m M_m|\psi\rangle \ ,$$

and the state of the system after the measurement is

$$\frac{M_m|\psi\rangle}{\sqrt{\langle\psi|{M^\dagger}_m M_m|\psi\rangle}} \ .$$

The measurement operators satisfy the completeness equation,

$$\sum_m {M^\dagger}_m M_m = I \ ,$$

where I is the identity matrix.

The completeness equation expresses the fact that probabilities sum to one:

$$1 = \sum_m p(m) = \sum_m \langle\psi| {M^\dagger}_m M_m |\psi\rangle \ .$$

Postulate 4: The state space of a composite physical system is the tensor product of the state spaces of the component physical systems. Moreover, if we have systems numbered 1 through n, and system number i is prepared in the state $|\psi_i\rangle$, then the joint state of the total system is $|\psi_1\rangle \otimes |\psi_2\rangle \otimes \cdots \otimes |\psi_n\rangle$.

B

Quantum Gates

Quantum gates are the basic building blocks of quantum circuits, analogous to classical logic gates used in classical digital circuits. They are used to manipulate the quantum states of qubits. Like classical gates, quantum gates operate on input qubits to produce output qubits. However, unlike classical gates, quantum gates can operate on qubits in superposition, which allows for more complex computations than classical gates.

A special detail has to be taken into account when talking about quantum state transformations, they have to preserve information, and that means that they have to be unitary (e.g. a classic gate such as the *AND* gate does not preserve information, since you cannot recover the input bits of the gate with the output bit only). This also implies that there is not a direct matching counterpart for each of the classic gates in the quantum world. Quantum gates work by applying a unitary transformation to the quantum state of the qubit. The most common types of quantum gates include the *Pauli* gates *(X, Y, and Z)*, the *Hadamard* gate, and the *CNOT* gate. The first ones are one-qubit gates, while the last one is a two-qubit gate. These gates can be combined to create more complex quantum circuits. An example on how a quantum gate is applied to transform the state of a qubit can be seen in equation (B.1), where *U* is the unitary operator (matrix) that represent the quantum gate and $|\psi\rangle$ is the wave function of the qubit. In this case a *Hadamard* gate is applied to a qubit in an eigenstate $|0\rangle$ to transform it into a superposition of equal probabilities of states $|0\rangle$ and $|1\rangle$.

$$U|\psi\rangle \rightarrow H|0\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\ 1\end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\ . \tag{B.1}$$

As with classic gates, there also exist a set of quantum universal gates. The universality of quantum computing refers to the fact that any quantum computation can be expressed as a sequence of quantum gates. This universal set of quantum gates can therefore be used to perform any quantum computation. The most commonly used universal set of quantum gates is a *CNOT* gate in conjunction with other one-qubit gates.

C

The Deutsch-Jozsa algorithm

This algorithm takes as input a black box function $f(x)|x \in \{0, ..., 2^n - 1\}$ and $f(x) \in \{0, 1\}$, it determines whether is constant (always returns 0 or always returns 1) or balanced with only one call to the function *f*. In contrast, to solve the same problem a classical algorithm would require in the worst case scenario 2^{n-1} calls. For the *Deutsch-Jozsa* algorithm to work *f* is guaranteed to be either constant or balanced (returns 0 for half of the possible inputs and 1 for the other half). The output of the quantum algorithm in a single evaluation depends on whether *f* is constant or balanced.

The intuition behind how this is done lies on superposition of the input qubits. By applying *Hadamard* gates to the input qubits, they are put into a superposition of all possible input values, then applying the unitary operator U_f (quantum homologous of *f*) to them, we can evaluate the function in all $x \in \{0, ..., 2^n - 1\}$ with just one call. Interference between the different input values causes the output to depend on the global properties of the function, rather than on the specific input value, obtaining the desired result by calling the function and measuring just once.


UAM
UNIVERSIDAD AUTONOMA
DE MADRID