Optimising Quantum Device Design with Machine Learning

Chih Jung Lee

Harris Manchester College

University of Oxford

A thesis submitted in partial fulfilment

of the requirements for the Degree of

Master of Engineering

Trinity 2025

Contents

List of Figures

List of Tables

1 Introduction

The introduction of the qubit made possible solutions to intractable problems in classical computation [1]. Unlike a classical binary bit, a qubit $|\psi\rangle$ can be described using the Bloch Sphere through a linear combination of two orthogonal vectors $|0\rangle = \binom{1}{0}$ and $|1\rangle = \binom{0}{1}$:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1}$$

where α and β are the probability amplitudes subject to the $|\alpha|^2 + |\beta|^2 = 1$ constraint [2]. This is a result of quantum mechanical superposition, in which systems can exist in multiple states simultaneously. The key advantage of this is the immense computational power it provides: just N qubits can hold as much information as 2^N classical binary bits.

This exponential improvement has the potential to massively benefit humanity across countless domains including climate modelling, cybersecurity, drug discovery, and materials simulations. Fittingly, enormous effort has been poured by academia and industry to create application-ready quantum computers with high numbers of qubits and fault tolerance. Of the many methods that have been proposed to realise this power, the quantum dot (QD) has emerged as a promising candidate due to its compatibility with semiconductor manufacturing processes and potential for scalability [3].

On a high level, QDs are nanoscale structures that function as “three-dimensional boxes” used to confine electrons [2]. When confined in the QD, the electron receives a quantised energy spectrum, and its spin state characterises the qubit that is measured [4]. Specifically, this thesis focuses on electrostatically-defined QD devices, where QD formation is achieved by external electrostatic potentials applied through gate electrodes. These gate electrodes can be tuned in terms of their properties (including electrostatic potential, 2D positioning, 2D and 3D geometries) to facilitate QD formation and qubit control [4]. These parameters, in addition to the number of gate electrodes for a given QD, define the challenge of designing optimal gate electrode layouts for QD devices.

The design and tuning of gate electrodes has historically been based on iterative approaches; expert researchers employ their intuition and experience to manually tune devices, adjusting multiple gate electrode voltages to achieve desired electronic configurations [5]. This exploratory method has been invaluable in demonstrating the viability of QD qubits and learning more about their physics, revealing critical relationships between design parameters and qubit performance.

However, the process is tedious, expensive, and often leads to complicated gate geometries that increase the manufacturing complexity. Furthermore, current designs do not guarantee optimal layout design. Overall, this paradigm hinders the advancement of QD technology toward practical quantum computing applications. In ignoring the manufacturing problem, it unwittingly limits optimality and scalability, features that will become increasingly important as quantum computing matures.

Specific to the problem of QD devices, reinforcement learning offers a potential solution as it can run through different iterations of gate electrode layouts faster than trial-and-error while learning an optimal strategy for tuning which can be applied en-masse once trained. Moreover, its sheer scale of exploration affords reinforcement learning’s use in testing the feasibility of limiting electrode gates to simple geometries in consideration of the manufacturing task. This method presents the opportunity of standardising a fast, optimal, and manufacturing-first design philosophy, a big step towards the future of practical quantum computers. Working towards this goal, this thesis investigates the use of ***physics-based reinforcement learning to optimally design simple geometry gate electrode layouts***.

2 Literature Review

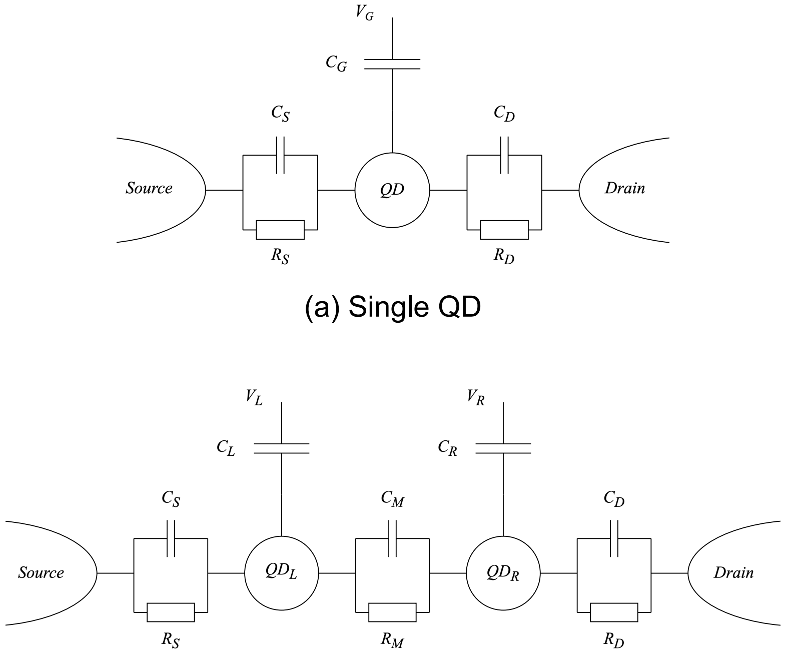
2.1 Quantum Dot Devices

Alluded to in the previous section, a QD is a static potential well confined in 3D that traps electrons for qubit measurement. In the context of QD devices, employing the 2 dimensional electron gas (2DEG) model allows for more convenient discussions. 2DEG is the most important low-dimensional system for transport to the long mean free path of electrons at low temperature, and this is achieved through a combination of strain engineering and modulation doping in semiconductor heterostructures [6]. On the 2DEG, electrons are free along the plane of the 2DEG surface, but are confined in the third direction. Formation of a QD on a 2DEG is verified by inspecting for large concentrations on a map of electrostatic potentials. The total electrostatic potential for every point in the 2DEG is modelled as the sum of components:

$$\varphi_{tot}(r) = \varphi_g(r) + \varphi_d(r) + \varphi_s(r) + \varphi_e(r) \tag{2}$$

where the electrostatic potential contributions are gate potentials φ_g from the gate electrodes following the pinned surface model, the disorder φ_d from imperfections of random stuck electron charges during cooling (disorder), the surface potential φ_s from the Fermi energy of the device, and electronic potential φ_e from the presence of charges in the 2DEG, each term a function of position on the device. This thesis focuses on silicon-germanium (SiGe) heterostructures, and in this context, φ_d is negligible and φ_e comes from holes, not electrons.

With reference to Figure 1a, the single QD is connected to source and drain contacts via tunneling barriers. Each barrier can be modelled as a parallel connection between a tunneling capacitor and a tunneling resistor: C_S with R_S for the source and C_D with R_D for the drain. The QD is also connected to a gate electrode via capacitance C_G, and the electrode provides an overall control mechanism for the tunneling of electrons into the QD via gate potential V_G [7].

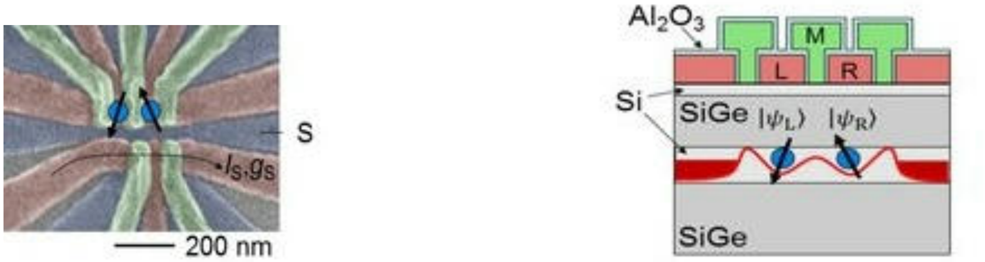


(a) Single QD

(b) Double QD

Figure 1: Networks of tunneling resistors and capacitors in QD architectures.

This model can be extended to architectures that capture more qubits, such as the double quantum dot in Figure 1b. In this scenario, the coupling between the QDs can be characterised by interdot capacitance C_M and resistance R_M. Gate electrode connections are similar to the previous case, each QD being connected via a capacitance. In practical systems, more gate electrodes are utilised to effectively control qubits.



(a) Scanning electron microscope (SEM) image of a Si/SiGe double QD device

(b) Cross-sectional view highlighting QD positions

Figure 2: Si/SiGe double quantum dot device.

A functioning architecture will have well-defined QD locations as shown in Figure 2a, verified by distinct qubits captured in Figure 2b [8]. Therefore, the electrostatic potential map on the silicon substrate serves as a useful premise for QD formation performance measurement.

2.2 Gate Electrodes

2.2.1 Gate Types

Gate electrodes are fabricated as metallic layers (typically aluminium or polysilicon) deposited on top of the heterostructure stack that can control and shape the electrostatic potential landscape within the semiconductor heterostructure [8]. In practical QD devices, gate electrodes are usually classified into three common types: plunger, barrier, and accumulation/reservoir. Their differences lie in size and voltage, with typical values listed in Table 1 [9], [10]. The limit of 4V on each gate is found empirically and serves to prevent leakage.

Table 1: Typical properties of different gate electrode types.

Gate Type	Typical Properties		
	Width	Voltage Range	Max/Stress Voltage
Plunger	100 - 150 nm	0.5 - 2 V	4 V
Barrier	30 - 50 nm	0.5 - 2 V	4 V
Accumulation/Reservoir	100 - 200 nm	1 - 2 V	4 V

Plunger gates are typically larger electrodes placed directly above the intended QD region. Their primary function is to tune the electrochemical potential of the dot, thereby controlling the number of electrons or holes confined within it. By adjusting the voltage on the plunger gate, researchers can load or unload single charges and finely control the dot's energy levels. The ideal scenario involves a 1:1 relation between the number of QDs and the number of plunger gates above the heterostructure. This is essential for initialising, manipulating, and reading out spin qubits in quantum computing.

Barrier gates are narrow electrodes positioned on either side of the plunger gate. They create tunable tunnel barriers between the QD and the source/drain reservoirs or between adjacent QDs. By varying the voltages on these gates, the tunneling rates can be precisely controlled, allowing for the isolation of the QD or the coupling between QDs for two-qubit operations. The interplay between plunger and barrier gates enables the formation of well-defined, controllable QDs with high-fidelity charge and spin manipulation.

Accumulation/Reservoir gates are used to accumulate carriers in the source and drain reservoirs adjacent to the QD. They ensure a steady supply of electrons or holes to the device and can help in tuning the chemical potential of the reservoirs relative to the QD.

2.2.2 QD Architecture

Si/SiGe heterostructures are widely used for QD devices due to their high electron mobility, compatibility with silicon fabrication, and tunable quantum confinement [8]. These devices typically employ a 2DEG formed at the interface of a strained Si quantum well sandwiched between SiGe barriers as illustrated in Figure 2b. Electrostatic gates patterned above the heterostructure in Figure 2a define QDs by depleting carriers in specific regions.

The architecture of a QD device in Si/SiGe heterostructures is defined by the spatial arrangement and functional interplay of gate electrodes. The primary choice is a linear arrangement of gates which involves two barrier gates flanking a central plunger gate, with optional accumulation/reservoir gates further out. Figure 2a demonstrates the use of this in the double QD case where the barrier-plunger-barrier configuration is concatenated twice on the top half of the SEM image to achieve two QDs, while the larger accumulation/reservoir gates lie on the far ends of this arrangement. This also works for the single QD case.

To deal with the capacitive crosstalk between gate electrodes that is not reflected in Figure 1 [11], practical systems typically require more than just a barrier-plunger-barrier configuration to realise a QD (not always 1:1 correspondence). Ultimately, it is the complex electrostatic interactions among different gate electrodes based on gate potential and gate positioning that define the properties of the QD formed.

2.3 Multi-Agent Reinforcement Learning

Reinforcement learning is a paradigm in machine learning where an agent learns optimal behaviour through trial-and-error interactions with an environment [12]. It relies on reward signals and not labelled datasets for learning (unlike supervised learning), and has explicit feedback in the form of rewards, penalties, and predefined goals (unlike unsupervised learning). This paradigm is most suitable for the task of designing optimal gate electrode layouts due to the need for physics-based feedback, but lack of sufficient "correct" and optimal layouts.

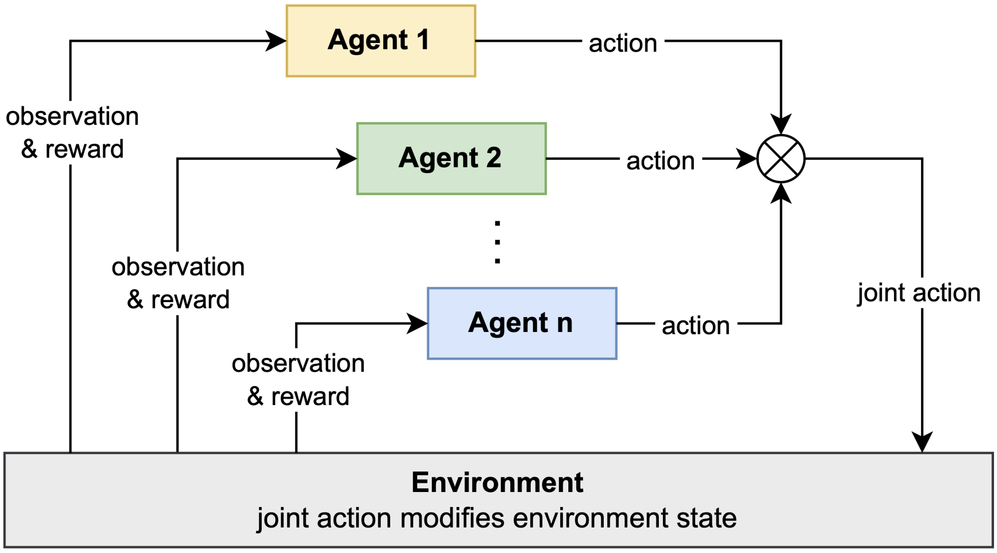


Figure 3: Schematic of multi-agent reinforcement learning.

By treating each gate electrode as an agent, the problem can be modelled using multi-agent reinforcement learning (MARL), in which optimal strategies are learnt for a set of agents in a multi-agent setting [13]. Figure 3 highlights the main differences of this model from single-agent reinforcement learning, notably the joint action which is a combination of the individual actions by different agents, and the separate observation-reward pairs that inform each agent of its next best action [14]. Over multiple iterations or *episodes*, agents begin to learn strategies that map states to actions, otherwise known as *policies*.

To fully specify solutions to the problem, it is useful to consider the type of *game model* that would most accurately describe the layout design problem. In reinforcement learning, game models can be organised hierarchically based on the number of agents, the number of states present, and the level of observability for these states by every agent as demonstrated in Figure 4 [14]. The layout design problem can be described as having n agents (gate electrodes) and m states (potential maps). These features already categorise the problem in the outer two sets of Figure 4: *stochastic games* and *partially observable stochastic games*, where observability refers to what individual agents can observe about their environment [14]. Assessing the level of observability would help decide which of the two sets best fits the problem.

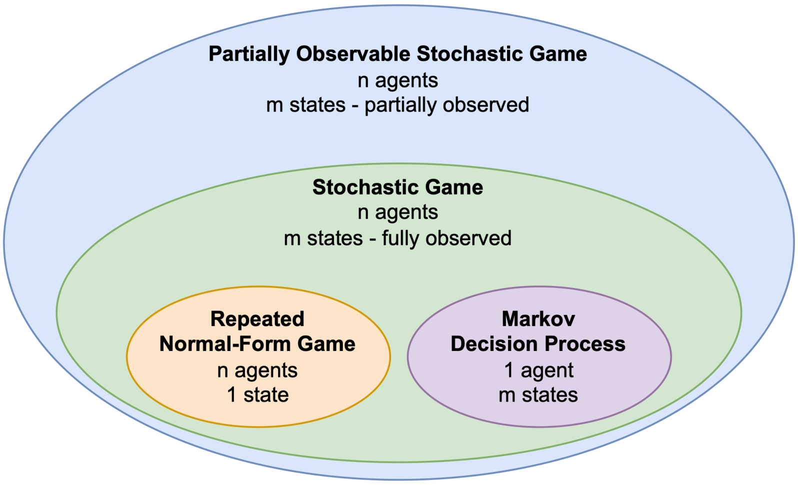


Figure 4: Hierarchy of game models.

At its crux, states in the design problem are largely defined by the electrostatic interactions between gate electrodes [4]. These can be modelled using physics in a way that is unobstructed to all gate electrodes. However, a view of the entire electrostatic map is unnecessary given that each gate electrode only exerts an influence in its locality. Another consideration is the knowledge of other agents' actions and rewards. Though trivial (since practical single QD designs only include < 10 gate electrodes), only neighbouring gate electrode information is necessary for a gate electrode to decide on the optimal action to take. This means that only a partial view of the full environment is necessary for every gate electrode, so the design problem can be more accurately modelled by a partially observable stochastic game.

2.3.1 Partially Observable Stochastic Games

The overall stochastic game model was introduced by Shapley to address the need for a general framework that could model dynamic interactions among multiple decision-makers where the state changes in response to the players' choices [15]. As mentioned previously, the most suitable game model to tackle the layout design problem with is the partially observable stochastic game. To achieve this, the stochastic game model needs to be extended using partial observations and a *finite-horizon*.

For M agents, a finite-horizon partially observable stochastic game is defined by a set of states S describing a finite set of possible configurations of all agents, a set of actions A_1, …, A_M and a set of observations O_1, …, O_M for each agent over a fixed number of training episodes [16]. Each agent i uses a stochastic policy π_θ : $O_i \times A_i \mapsto [0,1]$ to choose an action, which produces the next state following the state transition function T : $S \times A_1 \times \ldots \times A_N \mapsto S$. Each agent i gets a reward which is a function of the state and the agent's action r_i : $S \times A_i \mapsto \mathbb{R}$, as well as a private observation correlated with the state o_i: $S \mapsto O_i$. The goal for each agent i is to maximise its own total expected return, which is defined as:

$$R_i = \sum_{t=0}^{\mathrm{T}} \gamma^t r_i^t \tag{3}$$

where γ is a discount factor that informs the amount of importance placed on future rewards (e.g. $\gamma = 0 \rightarrow$ future rewards are not considered at all) and T is the time horizon. Since the time horizon is finite in a finite-horizon game, a discount factor $\gamma = 1$ is allowed as the return R_i is a finite sum for each agent i, meaning no divergence.

2.3.2 On-policy versus Off-policy

Reinforcement learning hinges on the interplay between exploration and exploitation, guided by policies that dictate an agent's behaviour as previously described. Two foundational paradigms govern how policies are updated: *on-policy* and *off-policy* learning [12].

In on-policy learning, the agent strictly refines its policy π using data generated by the same policy. This means that every update reflects the agent's current strategy, including its exploration mechanisms (e.g. ε-greedy action selection). The framework's defining feature is its alignment, where the behaviour policy (which collects data) and the target policy (which is optimised) are identical.

The benefit of such methods is that they are inherently stable; by relying on data from the current policy, they avoid distributional shift where outdated or mismatched data leads to divergent updates. Additionally, on-policy approaches naturally account for exploratory actions, ensuring that the policy's exploration strategy is baked into its learning process. However, they suffer from sample inefficiency, as they need to discard data from prior policies, constantly refreshing their interactions with the environment. Furthermore, their reliance on the current policy's exploration limits their ability to learn from diverse or suboptimal historical data, which can slow convergence. In MARL, on-policy methods ensure that agents adapt synchronously, as policies evolve using data from the current joint strategy. However, scalability suffers as coordination demands grow.

Off-policy learning, unlike on-policy, decouples the behaviour policy from the target policy. This separation enables agents to learn from diverse data sources, including exploratory policies or other agent's experiences. By reusing historical data (e.g. through replay buffers), they drastically reduce the need for real-time interaction, making them ideal for applications where data collection is costly. They also support flexible exploration where agents can adopt aggressive exploration strategies without compromising the target policy's optimisation. In MARL, this decoupling becomes especially powerful as agents can learn from others' behaviours or past interactions. However, they are prone to instability. Learning from mismatched data distributions introduces variance, particularly when the the behaviour policy poorly covers the target policy.

These tradeoffs directly shape algorithms like Multi-Agent Proximal Policy Optimisation (on-policy) and Deep Q Network (off-policy), as explored in the subsequent sections, which operationalise these frameworks in cooperative and competitive multi-agent environments.

2.3.3 Multi-Agent Proximal Policy Optimisation

Proximal Policy Optimisation (PPO) is a state-of-the-art on-policy algorithm that addresses the instability of traditional policy gradient methods. It constrains policy updates to prevent drastic changes. PPO's innovation lies in alternating between sampling data through interacting with the environment and optimising a clipped surrogate objective function shown in Equation 4 which limits the policy update magnitude using stochastic gradient descent [17].

$$L(\theta) = E\Big[\min\Big(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon) * \hat{A}_t\Big)\Big] \tag{4}$$

where $r_t(\theta)$ is the probability ratio of the new policy to the old policy $\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{old}}(a_t \mid s_t)}$, $\hat{A}_t$ is the advantage function at timestep t, and ε is a hyperparameter that controls the range of allowed policy changes.

Multi-Agent Proximal Policy Optimisation (MAPPO) is an extension of the single-agent Proximal Policy Optimisation (PPO) algorithm tailored for MARL. It trains two separate neural networks: an actor network with parameters θ and a critic network with parameters φ [18]. A recurrent neural network (RNN) implementation of MAPPO is shown in Algorithm 1, which assumes that all agents share the critic and actor networks. The critic network V_φ maps $S \longrightarrow \mathbb{R}$ and the actor network π_θ maps $O \longrightarrow A$.

Algorithm 1: Multi-Agent Proximal Policy Optimisation

```
Initialise $\theta$ and $\varphi$
Set learning rate $\alpha$
while $step \leq step_{max}$ do
    set data buffer $D = \{\}$
    for i = 1 to batch_size do
        $\tau$ = [ ] empty list
        initialise $h_{0,\pi}^{(1)}, ..., h_{0,\pi}^{(n)}$ actor RNN states
        initialise $h_{0,V}^{(1)}, ..., h_{0,V}^{(n)}$ actor RNN states
        for t = 1 to T do
            for all agents a do
                $p_t^{(a)}, h_t^{(a)} = \pi\left(o_t^{(a)}, h_{t-1,\pi}^{(a)}; \theta\right)$
                $u_t^{(a)} \sim p_t^{(a)}$
                $v_t^{(a)}, h_{t,V}^{(a)} = V\left(s_t^{(a)}, h_{t-1,V}^{(a)}; \varphi\right)$
            end for
            Execute actions $u_t$, observe $r_t, s_{t+1}, o_{t+1}$
            $\tau \longleftarrow \tau + \left[s_t, o_t, h_t^{(\pi)}, h_t^{(V)}, u_t, r_t, s_{t+1}, o_{t+1}\right]$
        end for
        Compute advantage estimate $\hat{A}$ via GAE on $\tau$, using PopArt
        Compute reward-to-go $\hat{R}$ on $\tau$ and normalize with PopArt
        Split trajectory $\tau$ into chunks of length $L$
        for $l = 0, 1, ..., \frac{T}{L}$ do
            $D = D \cup \left(\tau[l : l + T], \hat{A}[l : l + L], \hat{R}[l : l + L]\right)$
        end for
    end for
    for mini-batch k = 1, …, $K$ do
        $b$ ← random mini-batch from $D$ with all agent data
        for each data chunk $c$ in mini-batch $b$ do
            update RNN hidden states for $\pi$ and $V$ from first hidden state in data chunk
        end for
        Adam update $\theta$ on $L(\theta)$ with data $b$
        Adam update $\varphi$ on $L(\varphi)$ with data $b$
    end for
end while
```

In this algorithm, PopArt is an adaptive normalisation technique that is applied to normalise the targets used in the learning updates [19]. Its two main components are:

- **Adaptively Rescaling Targets (ART)**: to update scale and shift such that the target is appropriately normalised, and
- **Preserving Outputs Precisely (POP)**: to preserve the outputs of the unnormalised function when changing the scale and shift.

2.3.4 Deep Q-Networks

Deep Q-Networks (DQN) are convolutional networks trained using Deep Q-Learning which is an off-policy algorithm that uses a replay buffer to store experiences and use them to update the overall network [20].

Algorithm 2: Multi-Agent Deep Q Network

```
Initialise Q-networks $\theta_i$ for each agent $i$
Initialise target Q-networks $\theta_i' \leftarrow \theta_i$ for each agent $i$
Initialise replay buffer $D$
for episode = 1 to $M$ do
    Initialise environment and receive initial state $s_1$
    for t = 1 to max_episode_length do
        for each agent $i$ do
            Select action $a_i$ using $\varepsilon$-greedy policy from $Q_{i(s,a_i;\theta_i)}$
        end for
        Execute joint action $a = (a_1, ..., a_N)$, then observe reward $r$ and the next state $s'$
        Store $(s, a, r, s')$ in replay buffer $D$
        $s \leftarrow s'$
        for each agent $i$ do
            Sample random minibatch of transitions $(s, a, r, s')$ from $D$
            Set target $y = r_i + \gamma \max_{a_i'} Q_i'(s', a_i'; \theta_i')$
            Update $\theta_i$ by minimizing loss: $(y - Q_i(s, a_i; \theta_i))^2$
        end for
        Every $c$ steps, update target networks: $\theta_i' \leftarrow \theta_i$ for all $i$
    end for
end for
```

Previous work in the Quantum Device Lab (Natalia Ares) has shown that DQN can be used to solve the layout design problem when more complex gate electrode geometries are allowed. This algorithm learnt optimal policies for individual gate electrodes, which then performed the actions of removing/adding rows/columns to form a QD that is closest to the desired shape and position. This thesis aims to extend this work by adapting DQN to the multi-agent setting shown in Algorithm 2, and learning an overall optimal policy in a cooperative setting where the goal is to minimise the difference between the desired and actual QD shapes and positions.

3 Methodology

This thesis aims to specify MARL solutions to designing rectangular gate electrodes on top of the heterostructure stack. As a proof-of-concept project, the goal is to realise this for more feasible case of single QD devices.

3.1 Strategy

3.1.1 Layout Initiation

To apply MARL to the design problem, there has to be an initial layout to work with. By setting initial positions and gate voltages, an initial electrostatic potential map can be calculated via the physical modelling to then estimate the hole density of the layout.

The initial setup is established through 2D SEM images (in PNG format) that describe a 1400nm × 938nm window of the top view of the QD device. A scaling of 1.5px : 1nm is applied, meaning SEM images of dimension 934px × 625px are required. In tackling the single QD case, the initial layout involves the barrier-plunger-barrier configuration on the bottom half of the SEM image along with 2 accumulation gates that cut across the middle as illustrated in Figure 5.

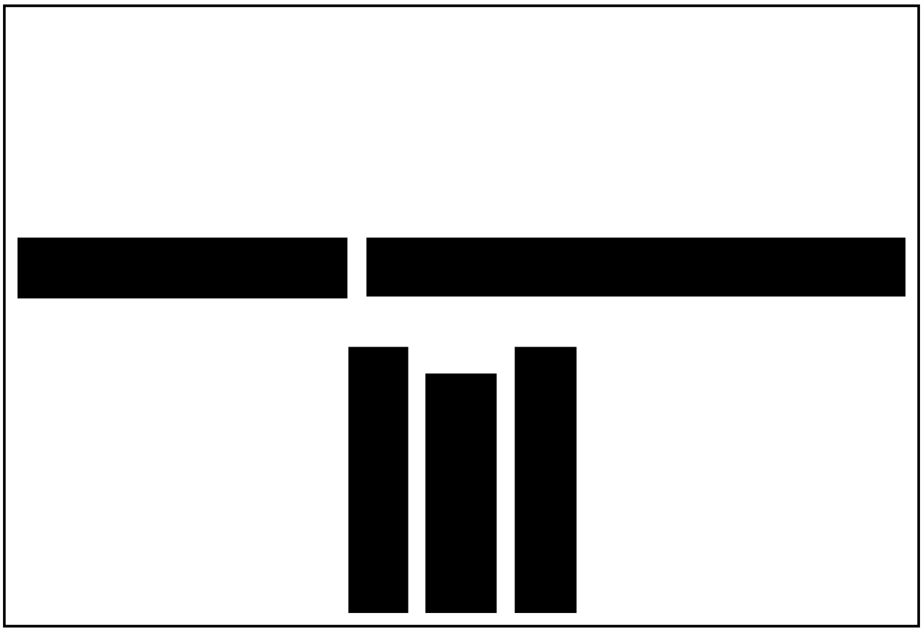


Figure 5: Initial gate electrode layout.

This layout is a combination of the individual gate electrodes. The input to the MARL algorithm would have each of these gate electrodes as separate SEM images. Notice that Figure 5 does not consider the electrometer gates whose role is in the measurement of changes in the QD rather than in the formation of the QD [21].

3.1.2 Action Space

From the perspective of each gate electrode, three types of actions can be performed.

1. *Translation* refers to the horizontal or vertical movement of each gate electrode.
2. *Scaling* is the expansion or contraction of the gate electrode.
3. *Voltage change* can also be applied to each gate electrode, referring to the increase or decrease of the gate potential.

To administer these actions, the action update algorithm needs to manipulate the gate electrodes based on the way they are represented computationally. In this problem, gate electrodes are fed into the algorithm as SEM images where for any one SEM image, the coloured pixels form one gate electrode. Due to the pixel representation, positional changes applied to gate electrodes require the removal or addition of rows or columns of pixels in the SEM images.

Through this observation, it is possible to simplify the action space by treating the translation and scaling actions as combinations of adding and removing rows or columns of pixels. This also solves the problem of translations that lead to spaces between the borders of the SEM image and the start of the gate electrode.

With this choice, the action space can be defined by the action space A where $|A| = 10$:

- add/remove left column,
- add/remove right column,
- add/remove top row,
- add/remove bottom row,
- increase/decrease gate potential.

3.1.3 State Estimation

The state of the system is the hole density distribution of the QD device. This is estimated by applying the physics model in Equation 2 to pixelated representation of the 934px $\times$ 625px SEM image. An initial state estimate is obtained by doing this for the inital gate electrode layout in Figure 5, and its hole density map is shown in Figure 6.

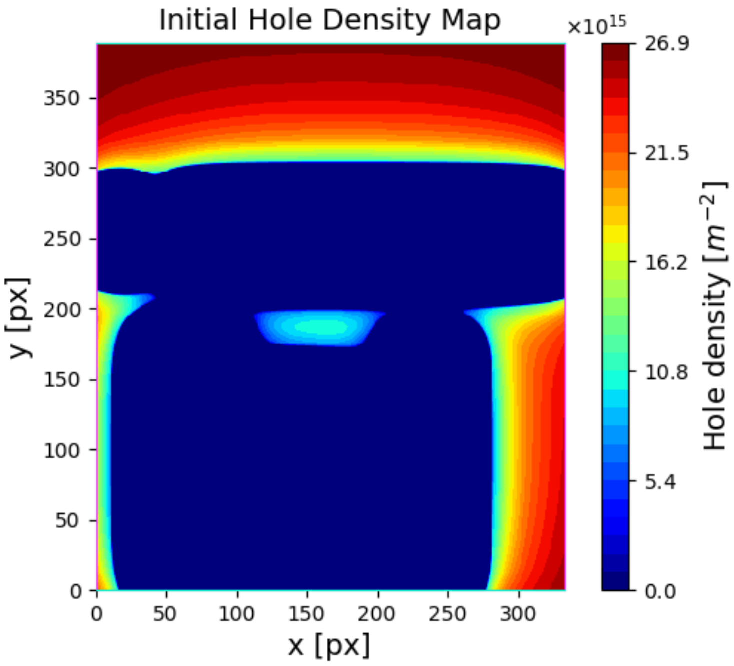


Figure 6: Initial hole density state.

For each state update, the physics model is used to recalculate the hole density distribution. While this sounds computationally expensive, the estimation algorithm is optimised through just-in-time compilation with JIT, meaning that estimation gets sped up to just $\sim$ 4 seconds per full state update. Furthermore, because of the nature of actions present in the action space, there is a large portion of the state space that does not change from one state to the next. For instance, the corners of the SEM image are not always affected by changes in gate electrodes. This allows for cached results to be used for subsequent state updates, tremendously speeding up the estimation algorithm to just $\sim$ 2 seconds per state update.

As mentioned in Section 2.3, it is unnecessary for a gate electrode to have a full view of the hole density map to decide on an action to take in the next time step. Instead, a gate electrode only needs to know the hole density distribution of the area within a certain radius of the gate electrode. As such, for every state update S, the hole density map in Figure 6 can be cropped into regions of 100px $\times$ 100px around the gate electrodes to serve as the observations $O_1, O_2, ..., O_5$. To include a cooperative aspect to the MARL reward, a crop of the QD itself (which lies close to the centre of the SEM image) should also be included. This need not serve as an observation for the gate electrodes, but will be processed using methods in Section 3.2 to provide additional scalar rewards to the gate electrodes during centralised training.

3.1.4 Constraints

So far, the description of the MARL algorithm has not considered any physical constraints. An important constraint is that gate electrodes must emerge from the borders of the SEM image. The reason for this is that this 934px $\times$ 625px window is ultimately a crop of the entire top layer of the QD device, and so gate electrodes actually extend beyond these borders to connect to voltage sources as suggested in Figure 1a.

There is also a constraint on the voltage applied to the gate electrodes. This is indicated in Table 1 where gate potentials cannot exceed 4V as doing so would lead to leakage and also threatens to damage the device. This constraint can be directly applied to the action space by clipping the voltage changes to between −4V and 4V.

Additionally, gate electrodes should not beallowed to overlap with each other as this would imply a short circuit in the device. To account for this, a Minkowski Sum can be used to ascertain that a minimum distance of 15px (10nm) is maintained between any two adjacent gate electrodes by forming new shapes for easy collision detection shown in Figure 7.

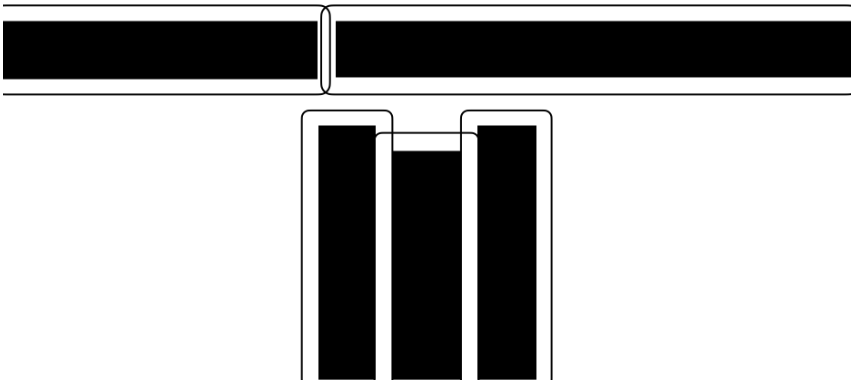


Figure 7: Full Minkowski Sum of initial configuration.

While Figure 7 shows the full Minkowski Sum of the initial configuration, collision checks are only done for one gate electrode at a time after it has been changed. That is why the algorithm would not need to do a Minkowski Sum for every gate electrode for every state update, and the Minkowski Sum needs to account for the full clearance of 15px as opposed to just half of it.

There are two properties of the layout problem description that helps to make collision detection even more efficient. Firstly, all gate electrodes are rectangular in shape. Secondly, only strictly vertical or horizontal electrode layouts are permitted. These two combine to make it sufficient for collision checks to be done only at the corners of the gate electrodes. As such, the Minkowski Sum can be adapted to look like Figure 8.

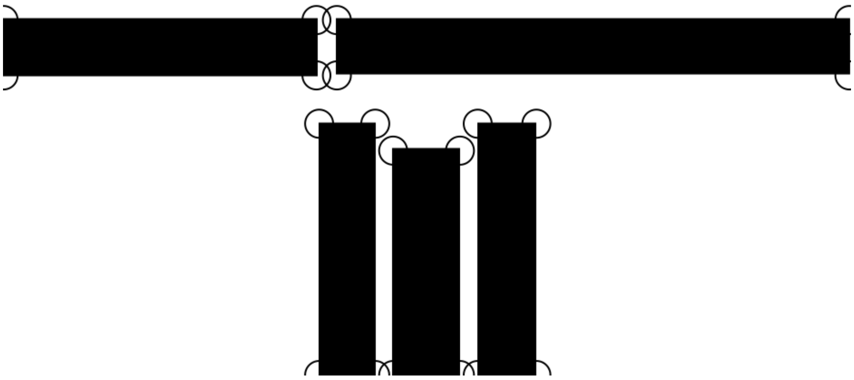


Figure 8: Reduced Minkowski Sum of initial configuration.

3.2 Performance Measure

A series of quantitative metrics are used to evaluate the quality of the QD formed, thus assessing the success of the MARL algorithm. These also help to inform the design of the reward function.

3.2.1 Hausdorff Distance

3.2.2 Mean Difference

3.2.3 Reward Function

3.3 Experimental Requirements

Due to the high costs associated with creating QD chips, this thesis only examines the feasibility of MARL methods through simulations.

4 Results

Cooperative evaluation, Performance metrics

5 Discussion

Indication of zero-sum between certain pairs of gate electrodes.

6 Conclusion

6.1 Applications

Designing QD devices

6.2 Age of Commercialisation

6.3 Future Work

7 Bibliography

[1] B. Schumacher, “Quantum coding,” *Physical Review A*, vol. 51, no. 4, pp. 2738–2747, Apr. 1995, doi: 10.1103/PhysRevA.51.2738.

[2] M. A. Nielsen and Chuang I. L., *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. doi: 10.1017/CBO9780511976667.

[3] A. I. Ekimov, A. L. Efros, and A. A. Onushchenko, “Quantum Size Effect in Semiconductor Microcrystals,” *Solid State Communications*, vol. 56, no. 11, pp. 921–924, Sep. 1985, doi: 10.1016/S0038-1098(85)80025-9.

[4] S. Harvey, “Quantum Dots / Spin Qubits,” *Oxford Research Encyclopedia of Physics*, Feb. 2022, doi: 10.1093/acrefore/9780190871994.013.83.

[5] C. Bureau-Oxton, J. Lemyre, and M. Pioro-Ladriere, “Nanofabrication of Gate-defined GaAs/AlGaAs Lateral Quantum Dots,” *Journal of Visualized Experiments*, no. 81, Nov. 2013, doi: 10.3791/50581.

[6] John H. Davies, *The Physics of Low-Dimensional Semiconductors*. Cambridge University Press, 1998.

[7] F.-M. Jing *et al.*, “Gate-Controlled Quantum Dots Based on 2D Materials,” *Advanced Quantum Technologies*, vol. 5, no. 6, 2022, doi: 10.1002/qute.202100162.

[8] T. Humble, H. Thapliyal, E. Munoz-Coreas, F. Mohiyaddin, and R. Bennink, *Quantum Computing Circuits and Devices*. 2018. doi: 10.48550/arXiv.1804.10648.

[9] M. Wolfe *et al.*, “Control of threshold voltages in Si/SiGe quantum devices via optical illumination,” *Phys. Rev. Appl.*, vol. 22, no. 3, pp. 34–44, Sep. 2024, doi: 10.1103/PhysRevApplied.22.034044.

[10] M. Meyer *et al.*, “Electrical Control of Uniformity in Quantum Dot Devices,” *American Chemical Society*, vol. 23, no. 7, pp. 2522–2529, Mar. 2023, doi: 10.1021/acs.nanolett.2c04446.

[11] W. Lawrie *et al.*, “Quantum dot arrays in silicon and germanium,” *Applied Physics Letters*, vol. 116, no. 8, Feb. 2020, doi: 10.1063/5.0002013.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. 2018.

[13] S. Albrecht and S. Ramamoorthy, “Comparative Evaluation of Multiagent Learning Algorithms in a Diverse Set of Ad Hoc Team Problems,” *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, 2019, doi: 10.48550/arXiv.1907.09189.

[14] S. Albrecht, F. Christianos, and L. Schafer, *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024.

[15] L. S. Shapley, “Stochastic Games,” *Proceedings of National Academy of Sciences of the United States of America*, vol. 39, no. 10, Oct. 1953, doi: 10.1073/pnas.39.10.1095.

[16] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, “Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments,” *Advances in Neural Information Processing Systems*, vol. 30, pp. 6380–6391, Dec. 2017, doi: 10.48550/arXiv.1706.02275.

[17] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” *arXiv*, Jul. 2017, doi: 10.48550/arXiv.1707.06347.

[18] C. Yu *et al.*, “The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games,” *arXiv*, Mar. 2021, doi: 10.48550/arXiv.2103.01955.

[19] H. van Hasselt, A. Guez, M. Hessel, V. Mnih, and D. Silver, “Learning values across many orders of magnitude,” *arXiv*, Aug. 2016, doi: 10.48550/arXiv.1602.07714.

[20] V. Mnih *et al.*, “Playing Atari with Deep Reinforcement Learning,” *arXiv*, Dec. 2013, doi: 10.48550/arXiv.1312.5602.

[21] X. Li, J. Sui, and J. Fang, “Single-Electron Transport and Detection of Graphene Quantum Dots,” *Nanomaterials*, vol. 13, no. 5, Feb. 2023, doi: 10.3390/nano13050889.

Department of Engineering Science

4YP Risk Assessment 2022

Description of 4YP task or aspect being risk assessed here: *(Read the Guidance Notes before completing this form)* Using Machine Learning techniques to optimise control electrodes in simulation		4YP Project Number: 13419
Site, Building & Room Number: Information Engineering Building (IEB), Level 3	Other Relevant risk Assessments: -	
Assessment undertaken by: Chih Jung Lee	Signed:	Date: 01 Nov 2024
Assessment Supervisor: Natalia Ares	Signed:	Date: 5 Nov 2024

Assessing the Risk*

You can do this for each hazard as follows:

- Consequences: Decide how severe the outcome for each hazard would be if something went wrong (i.e. what are the Consequences?) Death would be "Severe", a minor cut to a finger could be regarded as "Insignificant".
- Likelihood: How likely are these Consequences to actually happen? Highly likely? Remotely likely, or somewhere in between?
- Risk Rating: Start at the left of the coloured Matrix. On your chosen Consequences row, read across until you are in the correct Likelihood column for the hazard in question. For example, an outcome with Severe consequences but with a Low probability of actually happening equates to a Medium risk overall. In this case "Medium" is what should be written in the Risk.

RISK MATRIX		**LIKELIHOOD (or probability)**			
		High	**Medium**	**Low**	**Remote**
CONSEQUENCES	**Severe**	**High**	**High**	**Medium**	**Low**
	Moderate	**High**	**Medium**	**Medium/Low**	**Effectively Zero**
	Insignificant	**Medium/Low**	**Low**	**Low**	**Effectively Zero**
	Negligible	**Effectively Zero**	**Effectively Zero**	**Effectively Zero**	**Effectively Zero**

Overall statement of risk

- Carefully consider the risks associated with your project, the nature of the activity with which you will be engaged, and its location.
- Check the information from Health and Safety pages in the intranet including those specifically for the 4YP.

Students must discuss these risks with their supervisor.

☒ **Office work only**. My project involves only basic office work (paper and computers). It does not involve hands-on laboratory or field work of any kind. I am aware of the associated risks, including the health risks associated with the extended use of computers and display screens. No further assessment is required.

□ **Low Risk**. I consider the health and safety risks associated with my project to be low, working in alignment with existing risk assessments, I have referenced relevant risk assessments above and have agreed with my supervisor that no further assessment is required. For example, collecting data from existing systems within a lab.

□ **Medium Risk.** I consider there to be additional risks associated with my project as it requires risk assessment authorisation below:

Risk Assessments for Hazardous Substances & Biological Materials. The Biological & Chemical Safety Officer's (BSCO) signature is required for the final sign-off on Engineering Science COSHH Assessments. If the BCSO is unavailable the DSO can provide this signature. For IBME, the IBME Safety Officer can provide this signature. Reference E refers. The BCSO's signature is also required for risk assessments involving the use of biological materials.

Genetically Modified Organisms. Risk assessments involving genetically modified organisms require the BSCO's signature as well as approval from the Genetic Modification Safety Committee for the work to proceed. The department's Safety Policy refers.

Laser Risk Assessments: In addition to the supervisor of the laser equipment/experiment concerned, the Department Laser Safety Officer (DLSO) must also sign risk assessments involving lasers.

Where Specialist Safety Officers Originate Risk Assessments. Where the DSO or Specialist Safety Officers write, co-write or otherwise originate risk assessments they will be required to sign and authorize such risk assessments.

Requirements for review by specialists should be identfied within Safety Requirements section on https://fouryp.eng.ox.ac.uk/resourcetimepreview2.php

□ **High Risk**. This is a high risk activity as identified by Specialist Safety Officers.

Please review with Specialist Safety Officers where projects are Medium Risk sign below, ask your supervisor to countersign and then submit to Sharepoint site.

Signature of student: **Date: 01 Nov 2024**	**Signature of supervisor:** **Date:** 5 Nov 2024

Hazard *(potential for harm)*	**Persons at Risk**	**Risk Controls In Place** *(existing safety precautions)*	**Risk***	**Future Actions identified to Reduce Risks** *(but not in place yet)*
Slips and trips	**Staff and visitors to the IEB office**	**Good personal housekeeping. This means leaving no trailing leads or cables, keeping my work areas clear. It is also important to approach stairs to and from the office carefully to avoid trips.**	**Effectively Zero**	**Have a personal schedule for cleaning my work area. Develop a system to manage wires in and around my workspace.**
Extended use of electronic displays	**Users of the IEB office space**	**Workstation and equipment set to ensure good posture and to avoid glare and reflections on the screen. Work is planned to include regular breaks or change of activity.**	**Effectively Zero**	**Add light sources to personal workspace to allow for ideal lighting conditions when working. Include cushions to promote better posture when working.**

Hazard *(potential for harm)*	**Persons at Risk**	**Risk Controls In Place** (*existing safety precautions*)	**Risk***	**Future Actions identified to Reduce Risks** *(but not in place yet)*