

Automated Tuning and Optimal Control of Spin Qubits in Quantum Dot Devices

by

Andrija Paurevic

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Science
in
Physics (Quantum Information)

Waterloo, Ontario, Canada, 2024

© Andrija Paurevic 2024

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Silicon quantum dots present a promising foundation for realizing scalable quantum processors, leveraging the advantages of a mature semiconductor industry. Two significant challenges hinder their development: the laborious tuning of these devices and the coherent control of their spin qubits. This thesis presents contributions towards addressing these challenges by harnessing physics-informed machine learning.

Tuning these devices involves navigating complex parameter spaces, plagued with variability and fabrication imperfections, to identify optimal operating conditions. This process demands extensive time and resources by a researcher to perform large amounts of data collection and analysis. My work takes steps towards on achieving fully autonomous tuning of these devices, with the automated formation of a single quantum dot. This work involves the application of data analysis and computer vision techniques to extract relevant features from data, guiding the tuning process in real-time. This tool allows single quantum dots to be formed autonomously, freeing researchers to focus on investigating the physics of the device. Progress in multi-dot systems was also made by developing a data segmentation model that successfully identifies and segments charge and dot configurations in charge stability diagram data. This enables rapid data analysis to determine optimal voltage settings for achieving the desired device state.

Optimal control is crucial for guiding quantum systems through unitary operations while minimizing decoherence. Using a simulated open quantum system Hamiltonian for spin qubits, I developed a protocol to optimize experimental control signals, allowing for the implementation of unitary gate operations with arbitrary fidelity. The protocol designed experimental pulses for single-qubit rotations and entangling gates in a two-qubit system, achieving fidelities above the error correction threshold. Additionally, it utilizes modern machine learning frameworks, making it scalable to multi-qubit systems.

The work presented in this thesis serves as an important foundation for future advancements in our research group.

Acknowledgements

First and foremost, I would like to thank my supervisor, Jonathan Baugh, for his continuing support through my Master's. I would also like to thank all of the members of the Coherent Spintronics Group for being so hospitable and kind; always taking the time to answer my questions and share their knowledge. I am especially grateful to Floris Zwanenberg and the entire Nanoelectronics group at the University of Twente for providing our group with devices to measure. A special thank you goes to Tanmay Joshi for taking time out of his busy schedule to wire bond these devices for me and for keeping me company during testing.

Finally, I would like to thank my partner, Ana, for her unwavering support throughout my seven-year journey in academia. Her love and understanding has been my constant source of strength.

Overall, I believe this program has encouraged a wide variety of growth within me and I will always look back fondly on this experience. I cannot wait to apply all of the skills I've learned to my future career!

Dedication

I dedicate this work to my parents, who took the courageous leap of immigrating to Canada to provide me with opportunities they never had, and who have always supported me along the way.

Table of Contents

Author's Declaration	ii
Abstract	iii
Acknowledgements	iv
Dedication	v
List of Figures	x
List of Tables	xx
1 Introduction	1
1.1 Motivation	1
1.2 Outline	2
2 Semiconductor Quantum Dots	4
2.1 Semiconductors	4
2.2 Quantum Well Heterostructures	4
2.3 Quantum Dots	6
2.3.1 What is a Quantum Dot?	6
2.3.2 Device Architecture	6

2.3.3	Single Quantum Dots	8
2.3.4	Coupled Quantum Dots	14
2.4	Quantum System Dynamics	22
2.4.1	Open Quantum System Dynamics	22
2.4.2	Dynamical Maps	23
2.4.3	Markovian Processes	24
2.4.4	Master Equation	24
2.5	Quantum Information Processing	25
2.5.1	DiVincenzo's Criteria	25
2.5.2	The Qubit	26
2.5.3	State Initialization	26
2.5.4	Sources of Decoherence and Gate Times	26
2.5.5	Universal Set of Quantum Gates	27
2.5.6	Measurement	29
2.6	Summary	29
3	Machine Learning	30
3.1	Deep Learning	30
3.1.1	Automatic Differentiation	30
3.1.2	Parameter Optimization	32
3.1.3	Langevin Dynamics	34
3.2	Computer Vision Techniques	36
3.2.1	Edge Detection	36
3.2.2	Object Detection and Classification	38
3.3	Summary	42

4	Automated Tuning of Quantum Dot Devices	43
4.1	Motivation	43
4.2	Pillars of Automated Tuning	43
4.3	Data Segmentation Model	46
4.3.1	QFlow Dataset	46
4.3.2	Training Dataset	47
4.3.3	Results	50
4.3.4	Performance	55
4.3.5	Enhancements	55
4.3.6	Summary	55
4.4	Automated Tuning of a Single Quantum Dot	57
4.4.1	Device Under Test (DUT)	57
4.4.2	Measurement Set-Up	58
4.4.3	High-Level Overview	59
4.4.4	Forming a Single Quantum Dot	60
4.4.5	Bootstrapping	62
4.4.6	Coarse Tuning	65
4.4.7	Noise Characterization	72
4.4.8	Enhancements	73
4.4.9	Summary	73
5	Optimal Control of Semiconductor Spin Qubits	74
5.1	Motivation	74
5.2	QuDiPy Simulator	75
5.2.1	Hamiltonian	75
5.2.2	Lindbladian	76
5.2.3	Device Simulation	77
5.3	Control Loop Protocol	80

5.3.1	Overview	80
5.3.2	Signal Construction	81
5.3.3	Experimental Limitations	83
5.3.4	Evolution Propagator	85
5.3.5	Cost Function	88
5.3.6	Open System Pulse Design	88
5.4	Experimental Integration	97
5.4.1	Signal Chain Physics	97
5.4.2	Randomized Bench-marking	97
5.5	Enhancements	98
5.6	Summary	98
6	Conclusions and Outlook	99
	Letters of Copyright Permission	100
	References	104
	APPENDICES	111
A	Automated Tuning of Quantum Dot Devices	112
A.1	Single Quantum Dot Tuner	112
A.1.1	Bias Point Extraction	112
A.2	Coulomb Diamond Extraction	113

List of Figures

2.1	SiMOS heterostructure demonstrating the quantum well formation due to external electric fields below the Fermi level leading to an accumulation of a 2DEG s at the interface between Si and SiO ₂	6
2.2	Three dimensional quantum dot confinement in a SiMOS heterostructure. Voltages on the surface plunger (V) and barrier (W) gates directly enforce three-dimensional confinement forming the two quantum dots.	7
2.3	Scanning electron microscope image of a device with two quantum dot channels (top and bottom) separated by a screening gate. Adapted from Zwolak and Taylor [62] with permission. The lower channel confines four quantum dots located at their respective plunger gates ($P_{1\rightarrow4}$). The top channel confines two more quantum dots ($P_{5,6}$) that act as quantum dot charge sensors to detect lower channel activity and behaviour.	7
2.4	Single quantum dot modelled using the constant interaction capacitance model.	8
2.5	Impact of the plunger gate on the ladder of electro-chemical potentials. As the gate voltage changes it causes a shift in the ladder of electro-chemical potentials. Adapted from Hanson [17] with permission.	9
2.6	Coulomb blockade peaks as a function of gate voltage. The width of these peaks depends on the bias window applied across the quantum dot as well as the temperature of the device. Adapted from Hanson [17] with permission.	11
2.7	Schematic of a Coulomb diamond obtained from a bias spectroscopy measurement of a single quantum dot [51]. Areas outside the diamond exhibit current and inside the diamond is Coulomb blockade.	12

2.8	An example of a charge sensor with the gate voltage of the quantum dot fixed at a flank point (green or red). The Coulomb peak shifts due to a nearby disturbance in the electric field causing a noticeable change in current that can be recorded, indicating a perturbation.	14
2.9	A double quantum dot modelled using the constant interaction capacitance model.	15
2.10	Examples of charge stability diagrams for a double quantum dot. On the left represents two uncoupled quantum dots. In practice, there will be finite capacitive coupling between the dots leading to the more realistic charge stability diagram on the right. Adapted from Hanson [17] with permission.	17
2.11	Example of the features in a charge stability diagram demonstrating how strongly coupled quantum dots can merge together.	18
2.12	(a) Schematic of a charge stability diagram showing the gate voltage differences that should be used for calculating the capacitance of the network. (b) The presence of bias triangles due to a finite source drain bias across the device. Adapted from Penfold-Fitch [45] with permission.	20
2.13	(c,d) Double quantum dot charge stability diagrams before and after virtual gating. One can see that the charge stability diagram has been orthogonalized using the virtual gates (a,b) indicating one can isolate and address specific electro-chemical potentials without impacting nearby dots [36].	21
2.14	Schematic of a two level open quantum system the can spontaneously emit energy and transition energy states through spontaneous relaxation or excitation.	25
3.1	Example of a parameter network with inputs x , output y and ideal output $\hat{y}$. The goal is to find parameters, w and b , such that the objective function $\mathcal{F} = (y - \hat{y})^2$ is minimized. Local gradients (in purple) are calculated during the forward pass of the network and then by traversing the network backwards and computing the product of local gradients; parameter sensitivities (blue) can be found.	31
3.2	Algorithmic parameter update rule used by the Adam optimizer [26]. Adam employs many adjustments to the gradient to help improve optimization convergence and accuracy.	32

3.3	Gradient descent parameter optimization applied to a two parameter, highly non-convex, objective function $\mathcal{F}$. From the initial parameters $\vec{\theta}_i$, the optimizer follows the gradient (black arrow) towards the sub-optimal local minimum. Since the model strictly follows the gradient, it will remain at the local minima perpetually.	33
3.4	Langevin dynamics optimization applied to a two parameter, highly non-convex loss landscape. As the temperature of the system is initially hot, the optimization trajectory jumps around to explore the landscape. As it begins to cool, it exploits the gradient and is able to find the global minima.	35
3.5	(left) Example of a possible learning rate annealing schedule. It follows a cosine annealing schedule with warm restarts. This is beneficial as having a variable step size for gradient descent can help to speed up convergence and/or escape local minima. (right) Example of a temperature schedule that monotonically decreases with epoch number (for example $f(n) = T_{\max}/n + T_{\min}$). This prioritizes parameter exploration while the system is hot and switches to exploiting gradients when the system cools down.	35
3.6	A demonstration of how points in image space (left) are mapped to Sinusoidal curves in parameter space (right). The intersection of the sinusoidal curves represent lines in image space. The number of intersections corresponds to how likely those points correspond to a line in the image.	36
3.7	Hough transform applied to an image with two clear image lines [50]. Each point in the image (left) corresponds to a Sinusoidal curve (right) and the two clear intersection points correspond directly to the two lines seen on the image. Since the two lines are the same just rotated by ninety degrees, it explains why the two intersection points are seperated by ninety degrees as well.	37
3.8	Standard CNN architecture to detect an object in an image [46]. The image data is compacted through a series of computation layers into a probability vector where each element is the probability of a certain object being represented in the image. In this case, if the model is trained correctly, the first element would be one with the rest being zero.	38
3.9	Standard R-CNN architecture to detect an object in the image [1]. The image data is compacted through a series of computation layers into a vector where each element is the probability of a certain object being represented in the image as well as bounding boxes that surround the detected object.	39

3.10	Fast-RCNN architecture designed to detect an object in the image [14]. The image data is compacted through a series of computation layers into a vector where each element is the probability of a certain object being represented in the image as well as bounding boxes that surround the detected object.	40
3.11	Faster-RCNN architecture to predict the object classification and generate a bounding box for what is detected [13]. It does this by first training a region proposal network and then predict each region accordingly.	40
3.12	Mask-RCNN architecture designed to segment and classify certain cells in the test image [21].	41
3.13	An image processed using Detectron2’s Mask-RCNN model demonstrating its ability to segment and classify multiple objects in an image [57].	42
4.1	Example of a simulated image from the QFlow lite dataset [61]. The x and y axis are the plunger gate voltages defining each of the quantum dots. (left) Charge stability diagram measured from charge sensor data that will be the input to our object detection model. (right) State labels indicating the number of quantum dots at each voltage value: DD = Double Dot, SD = Single Dot, QPC = Quantum Point Contact, SC = Short Circuit.	46
4.2	Example of how QFlow charge sensor data and state labels (top) can be sent through the dataset creation pipeline to create one element of training data (bottom). The QFlow data is processed to five different class labels each with their own segmentation mask: CD = Central Dot, DD = Double Dot, L(R)D = Left(Right) Dot and ND = No Dot. Experimental data from QFlow had to be manually segmented since no state labels were provided with the data.	48
4.3	Example charge stability diagram from the training dataset that has been augmented with a counter clockwise rotation, cropping, and artificial noise. This aids in generating “artificial” data that closely resembles experimental data one would see from a device.	49
4.4	Trained coarse tuning model predicting dot configuration regions on an image from the validation dataset. On the left is the image with its true annotations and on the right is the same image passed through the model demonstrating a high degree of accuracy between predicted and actual segmentation masks and labels.	50

4.5	Trained coarse tuning model applied to three experimental images from the test dataset. In this case, no annotations were available, but one can see that the model predicts the dot configurations and their locations with a heuristically high degree of accuracy.	51
4.6	Trained charge tuning model applied to an image from the validation dataset. On the left is the image with its true annotations taken from QFlow. On the right is the same image inferenced through the model demonstrating a perfect match for expected labels and a high degree of accuracy for predicted segmentation masks. It might be hard to see but the labels are all correct compared to the true labels on the left.	52
4.7	Trained charge tuning model applied to the three available experimental images from the test dataset. In this case, no true annotations are available, but the labels for the masks are correct but the segmentation masks aren't perfect.	52
4.8	Example of a charge tuning path, parameterized by pulse length T , determined from the output prediction of a simulated image (top) and experimental image (bottom). Using the polygon centroids of the predicted charge state regions, and the desired charge path from the user, a path through pixel space can be calculated. Users can then convert their pixel path to a voltage path using the instrument step size used to measure the charge sensor data.	53
4.9	Example of a charge path from a simulated image (top) and experimental image (bottom) that has been virtually gated. This is done by using the slopes between neighbouring charge states to construct the virtual gating matrix seen in the image.	54
4.10	Top-view of the lateral gate defined geometry on the surface of the heterostructure. This schematic demonstrates the fan-out of gates from active region to macroscopic wire bonding pads. S and D are the source and drain Ohmic contacts respectively (responsible for setting the Fermi level and reading out the current), L is the lead gate (primarily responsible for 2DEG formation), LB/RB are the left and right barrier gates (used to deplete the 2DEG to confine a quantum dot) and P is the plunger gate (used to tune the electrochemical potential of the dot). The lateral area of the desired quantum dot region is designed to be $80 \times 80\text{nm}^2$	57

4.11	Side-view cross section of the device showing the heterostructure layering of materials as well as the relative position of the channel gates. The quantum dot would be ideally formed between the two barrier gates at the interface of the Si and SiO ₂ layer. The thickness of each layer is on the order of a few nanometres.	57
4.12	High-level overview of the measurement set-up required to execute the tuning protocol. (1) The software is run on the laboratory computer and communicates via (2) GPIB to all of the instruments. (3) The voltage source is connected through (4) the cryogenic fridge to (5) the device. The current is read out from the drain ohmic being connected to a (6) current-to-voltage pre-amplifier which is then connected to a (7) digital multi meter for readout.	58
4.13	The measurements required to tune a single quantum dot for an accumulation mode, n-type device. (1) By biasing the device and sweeping all of the channel gates together towards a positive value, a 2DEG is formed and the device <i>turns-on</i> . (2) The barriers defining the dot are tested on whether they can <i>pinch off</i> the current in the channel completely; this is done by individually ramping the barrier gates and measuring the current. (3) By performing a two-dimensional voltage sweep spanned by the barriers (around their pinch-off values), a bias point can be selected (gold star). This point realizes a central dot in the device. (4) If the bias point is successful, Coulomb oscillations will be visible upon sweeping the plunger gate voltage at this voltage configuration. (5) By sweeping the plunger and the device bias together, well-defined Coulomb diamonds should form. The diamonds can be processed to estimate dot properties. (6) Noise characterization can be done by looking at the power spectral density curve (optional).	60
4.14	High-level overview of the required measurements and data analysis performed in the tuning software. Any results from the data analysis can and will be used as input to the next measurement stage. If any error or anomaly is detected, the tuning software will abort and ground the device; from there a user can take over the tuning process and see what went wrong.	61
4.15	Global turn-on curve of the accumulation mode, n-type DUT. The voltage on all of the channel gates are increased to accumulate a 2DEG in the channel. The measurement was automatically terminated once the maximum current threshold of $I_{\max} = 1.5$ nA was measured. The data was then fit to a logarithmic curve to extract the turn-on properties, including turn-on and saturation voltage.	63

4.16	Pinch-off curves demonstrating each barriers ability to pinch-off the 2DEG that has been formed. The measurement is done by independently ramping each barrier gate voltage from towards zero. Each curve is fit to a sigmoid and the fit parameters are stored as metadata to be used later in the tuning process. The oscillations seen are due to quantum tunneling events when the device is in between the on and off state.	64
4.17	Two-dimensional I-V measurement used to determine the barrier voltages that realize a single central quantum dot. The Hough transform was used to detect Coulomb oscillations and a bias point along those lines was selected (black star). The domain of this scan came from the pinch-off widths of the barriers; as that is where quantum behaviour will occur.	66
4.18	Coulomb oscillations obtained by sweeping the plunger gate voltage with barriers fixed at a detected bias point. In this case we can see well defined, evenly spaced Coulomb peaks indicative of a quantum dot having been formed.	67
4.19	Processed Coulomb oscillations to extract flank points and sort them based on sensitivity (low, medium or high).	68
4.20	Measurement data showing the formation of Coulomb diamonds. As the plunger gate becomes more negative the channel becomes pinched off resulting in a diamond that won't close. Although, as the plunger gate becomes more negative, uniform, well-defined diamonds are formed.	69
4.21	(top) Detected Coulomb diamonds (0,1,2) from a cropped region of the measurement in Figure 4.20 that showed the clearest diamonds. From the geometry of each detected diamonds, device properties can be estimated informing the user on the characteristics of the quantum dot formed, as shown in Table 4.1. (bottom) Temperature estimation based on Coulomb oscillations taken at $V_{SD} = 100\mu V$. At this low-bias regime, the temperature of the charges can be estimated from fitting the data to their theoretical behaviour.	71
4.22	Current power spectral density for three different plunger gate sensitivities. As expected, the noise follows a $1/f$ spectrum when not at Coulomb blockade.	72
5.1	Example geometry of a double quantum dot SiMOS device with one spin qubit in each dot. External magnetic fields come from an ESR resonator near the device and the potential inside the heterostructure arises due to the voltages applied onto the surface gates ($\vec{V}$ and $\vec{W}$). Each of the quantum dots are assumed to have no more than one spin qubit each.	75

5.2	Demonstration of how voltage pulses are turned into effective pulses using the QuDiPy interpolation object. In this case, with such a high tunnel barrier voltage between the two qubits the electrons are well isolated and so have essentially zero exchange interaction. Being so well isolated, it explains why the deviation in g -factors so closely resemble the behaviour of the voltage pulses.	79
5.3	Flow diagram of the optimization protocol used for designing experimental control signals that maximize unitary gate fidelity. First, optimization parameters and system values are defined and initialized. The loop will then begin and will consist of signal construction, effective parameter interpolation, propagator construction and fidelity calculation. The parameters are then updated based on a modified Langevin dynamics update rule. This loop will continue until either the minimum fidelity or the number of iterations is met.	80
5.4	The three signal expansions the user can choose from. Each element in the parameter vector $\vec{\theta}$ will map to an individual signal expanded either using the (1) Gaussian, (2) PWC or (3) Fourier expansion. In this example, the value of the element in the vector corresponds to the amplitude of the basis signal that is used in the expansion. The resultant signal (green) is obtained by summing all of the individual expanded signals.	81
5.5	Example of a Gaussian signal being expanded and constrained in amplitude and bandwidth. (1) Each element in the parameter vector is converted to individual Gaussian signals (yellow and blue) which are then summed together to create the unconstrained signal (green). (2) Depending on the signal parameters, the resultant signal may exceed pre-defined experimental constraints and (3) is then subsequently constrained to get the final signal $\theta(t)$	83
5.6	The unitary gate operations that was used to test the performance of the pulse design protocol that accomplish (left) single qubit rotations with predominantly g -factor deviations and ESR field and (right) delicate exchange interactions.	89
5.7	Example showing how the controlled- Z gate can be constructed using just single qubit rotations and the $\sqrt{\text{SWAP}}$ gate.	89
5.8	Hadamard Gate: The infidelity, $1 - \mathcal{F}$, curve during the training process. It took just over twenty minutes to find a set of experimental signals that realize the gate to a fidelity above the error correction threshold.	91

5.9	Hadamard Gate: (left) The optimized experimental voltage gate pulses. (right) ESR amplitude and phase respectively.	92
5.10	Hadamard Gate: (left) Interpolated g -factor deviations for the two qubits and (right) the exchange coupling between the two qubits. As expected for a single qubit rotation little to no exchange interaction is required.	92
5.11	Hadamard Gate: State evolution of $ q_1q_2\rangle = 11\rangle$ with unitary gate infidelity of 10^{-3} . The states remain pure during the evolution as desired.	93
5.12	$\sqrt{\text{SWAP}}$ Gate: The infidelity, $1 - \mathcal{F}$, curve during the training process. The sensitive nature of the exchange interaction is evident in the oscillatory response of the curve.	94
5.13	$\sqrt{\text{SWAP}}$ Gate: (left) The optimized voltage pulses obtained. (right) The ESR amplitude and phase respectively. Exchange interactions don't require any ESR pulsing so they were set to zero for the duration of the optimization.	95
5.14	$\sqrt{\text{SWAP}}$ Gate: (left) Interpolated g -factor deviations for the two qubits and (right) the exchange coupling between the two qubits. As expected there is very little deviation in the g -factor for the duration of the pulse but a strong exchange interaction for around 100 microseconds.	95
5.15	$\sqrt{\text{SWAP}}$ Gate: Example initial triplet state evolved under the optimized voltage and ESR pulses of infidelity 10^{-3} . This initial state was chosen to showcase the effect of the unitary gate to "undo" the entanglement to return the expected state at the end of the pulse.	96
6.1	Reuse and Permissions License for Figure 2.3	101
6.2	Reuse and Permissions License for Figures 2.10, 2.6 and 2.5.	102
6.3	Reuse and Permissions License for Figure 2.12	103
A.1	(top-left) The measurement data is first processed by taking the gradient along a 45 degree line. (top-right) Edges are then extracted and (bottom-left) the lines present in the edges are detected. (bottom-right) Filtering any lines do not meet the criteria of Coulomb oscillations lets us choose the midpoint of said lines at the optimal bias points. In this case, only two bias points were suggested to the user.	113

A.2 (left) Raw measurement data with the detected diamonds overlaid on top. (right) By binary thresholding the measurement data (to extract regions of Coulomb blockade), clear diamonds begin to emerge. First, the image is cut in half horizontally and the lines for each half are determined (blue and red). By sorting all detected lines in terms of their half and slope sign, a diamond can be constructed by grouping the appropriate set of lines going from left to right in the image (upper positive + upper negative + lower positive + lower negative = one diamond). The intersection points of all four lines are then found and grouped together to define the geometry of the diamond. . 114

List of Tables

4.1	Device properties extracted from the Coulomb diamonds measurement in Figure 4.20. The quantum dot area on the device should have a radius of $r \approx 40\sqrt{2}$ nm, indicating the dot formed is slightly larger than the device's intended design.	70
5.1	System parameters used for the optimization of the Hadamard and $\sqrt{\text{SWAP}}$ gates.	90
5.2	Hyper-parameters used for the optimization of the Hadamard and $\sqrt{\text{SWAP}}$ gates.	90

Chapter 1

Introduction

1.1 Motivation

With computers becoming a technology that is used in every sector of human society, the need for high performance computing is self evident. Unfortunately, certain problems require computation levels that surpass the modern capabilities of classical computers. Scientists have been predicting this for almost half a century, leading to quantum computers emerging as a proposed method of computation that promises to not only match the computational efficiency of classical computers but eventually supersede them. Whereas classical computers are built from the logical manipulation of binary bits derived from large transistor networks, quantum computers are built from a collection of quantum bits (qubits) each of which can exist in a coherent superposition between zero and one. When two or more of these qubits interact, quantum entanglement can occur, further increasing the computational power of these machines. With the power of superposition and entanglement quantum computers exhibit powerful computational scaling; N qubits can store up to 2^N information states leading to an exponential relationship between the size of the system and its computational complexity.

There are a multitude of platforms available for building quantum bits, some of which harness existing phenomena in nature and some have been meticulously engineered by scientists. The platform of interest in this thesis, spin qubits in semiconductors, are an emergent platform showing promise in commercial quantum computing. These devices make use of the electron and its spin properties, which can be argued to be one of nature's perfect quantum bits (qubits). With its quantized spin of $\pm\frac{1}{2}$ and electric charge, it makes an excellent candidate for a qubit as its position in space and spin state can be manipulated

through external electromagnetic fields. Moreover, this platform of quantum computing leverages the ever growing support from condensed matter and solid-state physics research and the mature state of the semiconductor industry (thanks to consumer electronics).

The formation of spin qubits in these devices can be done with careful electrostatic control of metal gates deposited on their surfaces. In order to get the device into an operating regime, the device parameters must be heuristically manipulated, relying on experimental intuition and informed guesses by the researcher. Not only is this time consuming but it often leads to sub-optimal exploration and configuration of the device. With modern device designs requiring multiple gates to isolate and control just a single qubit, the heuristic tuning of these devices rapidly becomes intractable.

Even after the qubits have been formed, to coherently evolve their spin states in an attempt to perform various quantum computation protocols is very challenging. There are many sources of noise that actively decohere the spin state of the qubit; leading to a loss in quantum information. Therefore, a careful strategy to optimize the control of these qubits in the presence of experimental limitations is of great importance.

The work in my thesis is motivated by the overarching goal of scaling these devices by addressing two main issues: the automated tuning of these devices and the coherent evolution of their spin qubits. The former will be tackled by developing software to aid in the tune up of a device, whether it be automating certain repetitive measurements, or analyzing data with computer vision techniques. The latter will be addressed through the use of machine learning to design voltage gate pulses that optimize desired quantum gate operations in the presence of experimental limitations. While the projects do not necessarily build on each other they instead can be considered crucial stepping stones towards reaching large-scale semiconductor quantum processors.

1.2 Outline

This thesis is organized into two halves composed of four chapters; the first half is the relevant theoretical background covering semiconductor quantum dots and important machine learning techniques. The second half presents my original contributions addressing automated tuning and optimal control.

Background

Chapter 2 focuses on the fundamentals of silicon quantum dots and spin qubits. It begins with a background section covering semiconductor heterostructures and how quantum dots can be isolated in their conduction layer through modern lateral gate designs. This is then followed with an understanding of quantum dot behaviour that gives precedent to the standard set of measurements used for device tune-up and characterization. With an understanding of quantum dots, the theory behind open quantum systems is introduced which leads into quantum information processing with spin qubits.

Chapter 3 focuses on providing the necessary context for the machine learning techniques used throughout this thesis. This chapter is broken into two distinct sections: computer vision and signal optimization. As mentioned, computer vision is crucial to the automated tuning of quantum dot devices as various visual features in measurements are heuristically analyzed by researchers. Signal optimization is crucial for optimizing control pulses with a goal of performing coherent quantum computations against experimental limitations.

Contributions

Chapter 4 covers my personal efforts in advancing the automated tuning of quantum dot devices. The chapter opens with the discussion of the five pillars of automated tuning required to scale semiconductor quantum dot devices. My novel application of an existing Mask-RCNN model to quantum dot data is then introduced. In this section, I demonstrate the model's ability to address three important pillars of automated tuning: coarse tuning, charge tuning and virtual gating. This is followed by a demonstration of my automated tuning software by successfully tuning a single quantum dot in a single electron transistor device. The chapter concludes with a suggestion for future work as well as an outlook on this work.

Chapter 5 covers my personal efforts made to the optimal control of spin qubits in these devices. Realistic device geometries were simulated and an optimization control loop was developed that designs high fidelity spin qubit unitary gate operations. The work was implemented into our group's general-purpose Python package for semiconductor quantum dot simulations [16]. The chapter concludes with a discussion of next steps and how these pulses can be implemented on real devices.

Chapter 2

Semiconductor Quantum Dots

2.1 Semiconductors

Crystalline materials can be classified into one of three groups: (1) conductors, (2) insulators or (3) semi-conductors. There are two parameters that uniquely classify a given crystalline material, its bandgap and Fermi level. By definition, the bandgap of a material is the energy difference between the highest valence band state and the lowest conduction band state; essentially quantifying how much energy is required to excite a charge carrier from a valence bond to a free mobile carrier. The Fermi level of a material is defined as the energy at which the occupation probability of the next available state is one half. For conductors and semiconductors, the Fermi level would be at the exact energy center of the bandgap. However, for semiconductors we can tune the Fermi level by doping the material, allowing one to manipulate the conduction of charge carriers. External electric fields are also used to bend silicons conduction bands away from or towards the Fermi level in the material to modify its conductivity. Silicon is a familiar semiconductor material used in modern devices whose conductivity is modified by doping the material with either n-type (electron) or p-type (hole) charge carriers. By cleverly combining conductors, insulators and semi-conductors together, modern practical devices can be fabricated.

2.2 Quantum Well Heterostructures

A hetero structure is a material system that is composed of layers of two or more different semiconductors [20]. These layers are thin, often only a few nanometers thick, and

can be grown using various fabrication techniques. These structures are fundamental to quantum dot devices as they are what provide the required quantum confinement and electronic properties that allow us to manipulate and control quantum dot formation and manipulation.

Material Composition

There are two modern silicon heterostructures that are used to form quantum dots: Si/SiGe and SiMOS. For this thesis, we will focus on SiMOS, as it is of interest to our group for device fabrication. The robust manufacturing process of silicon semiconductor devices can be leveraged to construct devices that are able to form quantum dots each of which can hold single charges in them.

Band Alignment and Lattice Mismatch

With some deliberate heterostructure growth, the silicon layer can be placed under strain due to a lattice constant mismatch. This strain leads to significant deviation in the conduction band. The depth of this deviation can be adjusted by applying external electric fields; pulling it lower to accumulate charge carriers. This is known as the formation of a quantum well creating a two-dimensional charge-carrier gas in an accumulation mode device. Naturally, the charge carriers could either be electrons (2DEG) or holes (2DHG) depending on the doping of the material. This gas can be thought of as a sheet of charges that are *vertically* confined, but are free to move *laterally*. At very low temperatures and/or with strong vertical confinement, these electrons occupy the same vertically confined state but can be considered a laterally continuous density of states.

Figure 2.1 is a cartoon representation of such heterostructure and its band gap behaviour that leads to the formation of a 2DEG in modern silicon semiconductor quantum dot devices.

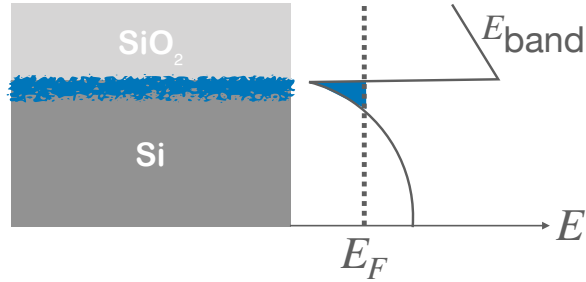


Figure 2.1: SiMOS heterostructure demonstrating the quantum well formation due to external electric fields below the Fermi level leading to an accumulation of a 2DEG s at the interface between Si and SiO₂.

2.3 Quantum Dots

2.3.1 What is a Quantum Dot?

The idea of quantum dots fundamentally boil down to the principles of quantum confinement. The 2DEG discussed earlier is an example of a quantum well which is confined to one dimension. Naturally, you can further confine the charges in another direction to construct what is known as a *quantum wire*. Confining the gas in the remaining direction results in a quantum dot.

This can be done by depositing metal surface gate electrodes and inducing a voltage configuration that results in quantum dot confinement. These gate electrodes are also separated from the heterostructure (and thus the 2DEG) by a layer of dielectric material to avoid any direct current flow. This results in a 'particle-in-a-box' quantization where the energy levels are discretized; producing an "artificial atom". A example of double quantum dot confinement can be seen in Figure 2.2.

2.3.2 Device Architecture

Due to fabrication limitations of surface gates, lateral confinement is typically much weaker than the vertical confinement. This results in typical lateral dot sizes on the order of tens to hundreds of nanometres. For the scope of this thesis, it is not crucial to details of device fabrication but instead to appreciate the purpose and behaviour of different device components. The gate electrodes fan out from the active region of the device to a

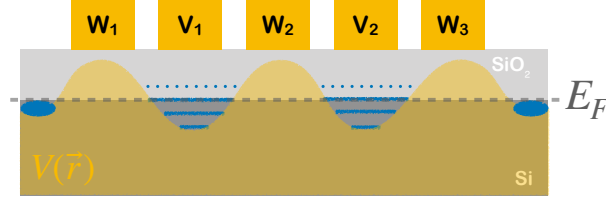


Figure 2.2: Three dimensional quantum dot confinement in a SiMOS heterostructure. Voltages on the surface plunger (V) and barrier (W) gates directly enforce three-dimensional confinement forming the two quantum dots.

macroscopic set of bonding pads. Nearly every quantum dot device will have the following: a set of ohmics, screening gate(s), and “finger” gate(s). Ohmics are metal contacts built into the 2DEG to allow us to set the Fermi level inside the quantum well, as well as enabling measurement of the current between ohmics. Screening gates are used to deplete local areas in the 2DEG where charge carriers should not exist; often to isolate certain sub-devices within the design. The finger gates allow us to create a confinement potential that results in the formation of quantum dots. There are typically two types of finger gates: plunger and barrier gates. Plunger gates are typically responsible for tuning the number of charges in each dot. Barrier gates are responsible for tuning the tunneling rate of charges between dots or the neighbouring reservoirs of charges. Figure 2.3 shows an example of a real device.

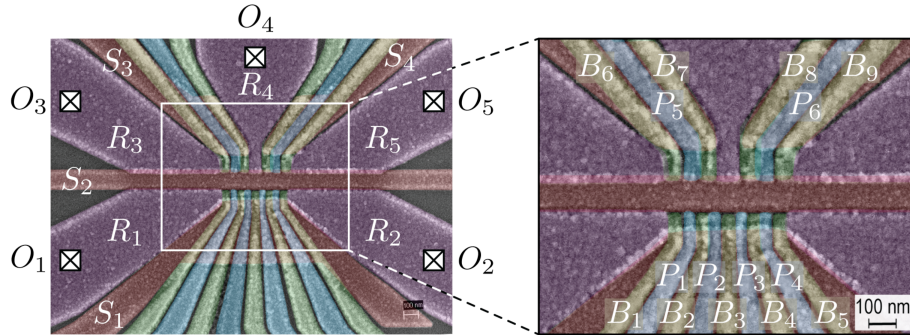


Figure 2.3: Scanning electron microscope image of a device with two quantum dot channels (top and bottom) separated by a screening gate. Adapted from Zwolak and Taylor [62] with permission. The lower channel confines four quantum dots located at their respective plunger gates ($P_{1 \rightarrow 4}$). The top channel confines two more quantum dots ($P_{5,6}$) that act as quantum dot charge sensors to detect lower channel activity and behaviour.

2.3.3 Single Quantum Dots

Constant Interaction Model

In practice, the quantum dots are complicated and difficult to model accurately. Fortunately, much of the important phenomena can be captured by representing the system as a simple capacitance model [17, 28]. This model is built on two assumptions: (1) the Coulomb interactions amongst and between electrons in the dot and the environment can be parameterized by a single, constant capacitance C_Σ and (2) the single-particle energy level spectrum is independent of these interactions. Surprisingly, this allows us to model quantum dot behaviour with a high-level of accuracy. We begin our analysis with a standard device geometry consisting of a *gate* nested between two *barrier* gates of which are then nested between a *source* and *drain* reservoir of charges (see Figure 2.4).

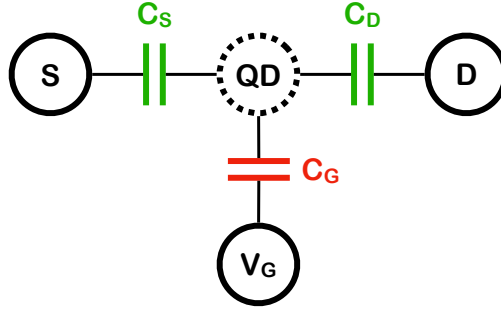


Figure 2.4: Single quantum dot modelled using the constant interaction capacitance model.

Charge Tunneling

We can begin by writing down the total energy $U(N)$ of the dot with N electrons in the ground state, with V_S, V_D and V_G representing the voltages applied to the source, drain and gate respectively,

$$U(N) = \frac{(-|e|(N - N_0) + C_S V_S + C_G V_G + C_D V_D)^2}{2C_\Sigma} + \sum_{n=1}^N E_n(B) \quad (2.1)$$

where $-|e|$ is the charge of an electron, B is the applied magnetic field, $E_n(B)$ is a sum over the occupied single-particle energy levels (depending on device confinement potential), $C_\Sigma = C_S + C_D + C_G$ due to the parallel network of capacitors and finally N_0 is the number of positive background charges originating from the hetero structure [17]. Using this energy equation, we can extract the electro-chemical potential, $\mu(N)$, which is defined as the energy required to add an electron to the quantum dot,

$$\mu(N) \equiv U(N) - U(N-1) \quad (2.2)$$

$$= \underbrace{\left(N - N_0 - \frac{1}{2}\right) E_C - \frac{E_C}{|e|} (C_S V_S + C_G V_G + C_D V_D)}_{\text{electrostatic}} + \underbrace{E_N(B)}_{\text{chemical}}, \quad (2.3)$$

where $E_C = \frac{e^2}{C_\Sigma}$ is defined as the charging energy of the dot. The reason why using the electro-chemical potential is superior to energy is because it depends linearly on the plunger gate voltage. This allows us to label the various states of the quantum dot as a ladder of electro-chemical potential values which can be *tuned* by changing the plunger gate voltage (see Figure 2.5) [17].

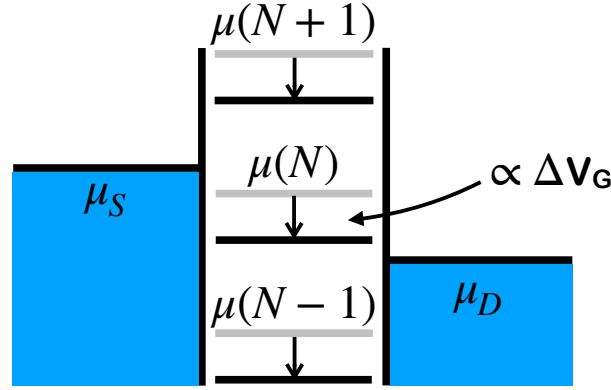


Figure 2.5: Impact of the plunger gate on the ladder of electro-chemical potentials. As the gate voltage changes it causes a shift in the ladder of electro-chemical potentials. Adapted from Hanson [17] with permission.

With the electro-chemical formulation, the addition energy is defined as,

$$E_{\text{add}}(N) = \mu(N+1) - \mu(N) = E_C + \Delta E, \quad (2.4)$$

where ΔE is the energy spacing between two discrete quantum levels (note that if the two quantum levels are spin degenerate, ΔE can be zero) [17]. This is important because we now know how much energy we need to match in order to get tunneling through the dot; ergo current! We can define a source drain *bias window* with V_{SD} ,

$$\mu_S - \mu_D = -|e|V_{SD}, \quad (2.5)$$

where current flow through the dot depends on aligning the electro-chemical potential in the dot inside *bias window*. As we adjust the dot gate voltage we can align the potential within the bias window in order for electrons to be able to tunnel from one reservoir, onto the dot, and then off to the empty states in the other reservoir. In Figure 2.5, we would have charge flow because the $\mu(N)$ energy state is between the bias window and so charges from the source reservoir can flow left to right. The chemical potential ladder picture of the quantum dot allows us to gain an understanding of its simplest, yet most fundamental phenomenon, Coulomb blockade.

Coulomb Blockade

The observation of Coulomb blockade in the I-V characteristics of a device indicates the formation of a quantum dot. On either side of the quantum dot we have reservoirs full of charge carriers which are filled to some Fermi level with a Fermi-Dirac distribution of occupied states. Energy levels (states) in the quantum dot well above the Fermi levels will have an occupation of zero, as there are no electrons available from the reservoirs to fill them. Conversely, energy levels well below this Fermi level will be occupied and essentially 'frozen' on the dot since all the states at the level in the reservoirs are already full. When the dot is in a state in which its energy level is not near the Fermi level, then current cannot flow through the dot. We say that the dot is exhibiting *Coulomb blockade*. The current through the dot will then exhibit Coulomb blockade peaks once a potential bias is placed across the dot as a function of gate voltage as seen in Figure 2.6. As one can imagine, the width of these peaks depends on two things: (1) the temperature of the device and (2) the bias window provided.

Coulomb Diamonds

We can further understand charge transport through a quantum dot using bias spectroscopy otherwise known as Coulomb diamonds [17]. If we look at Figure 2.5 one could also recognize that current flow through the quantum dot is possible not just by moving the

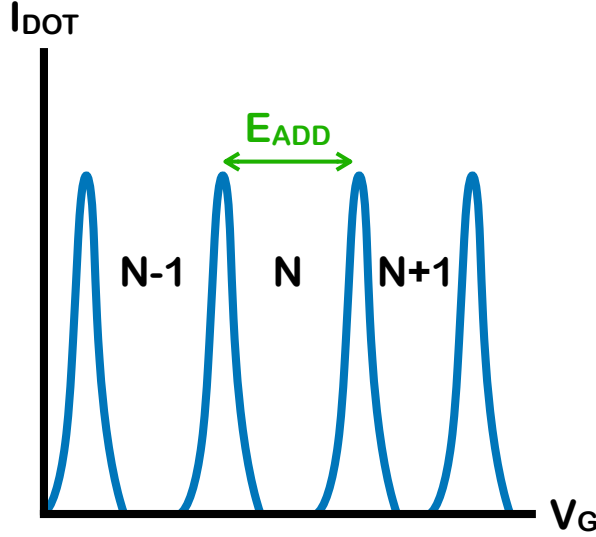


Figure 2.6: Coulomb blockade peaks as a function of gate voltage. The width of these peaks depends on the bias window applied across the quantum dot as well as the temperature of the device. Adapted from Hanson [17] with permission.

electro-chemical potential of the dot, but by also increasing the bias window. This gives us two degrees of freedom that directly have an effect on the current through the quantum dot. Therefore, by measuring the current through the quantum dot as a function of the gate voltage and source-drain bias, we will see diamond shaped regions emerging, where Coulomb blockade occurs in those regions (see Figure 2.7).

What is so powerful about Coulomb diamond measurements is that the capacitance model is embedded in the geometry of the diamond. This allows us to extract useful information about the quantum dot just by processing its geometrical properties. Assuming a grounded drain ohmic contact, the shift in electro-chemical potential comes from changes in the gate and source voltage. The positive and negative slopes of the diamond in Figure 2.7 directly depend on the capacitance of the dot to the drain and source respectively [17],

$$\beta = \frac{C_G}{C_G + C_D}, \quad \gamma = \frac{C_G}{C_S}, \quad (2.6)$$

The diamonds have a height and width directly dependent on the total capacitance of the

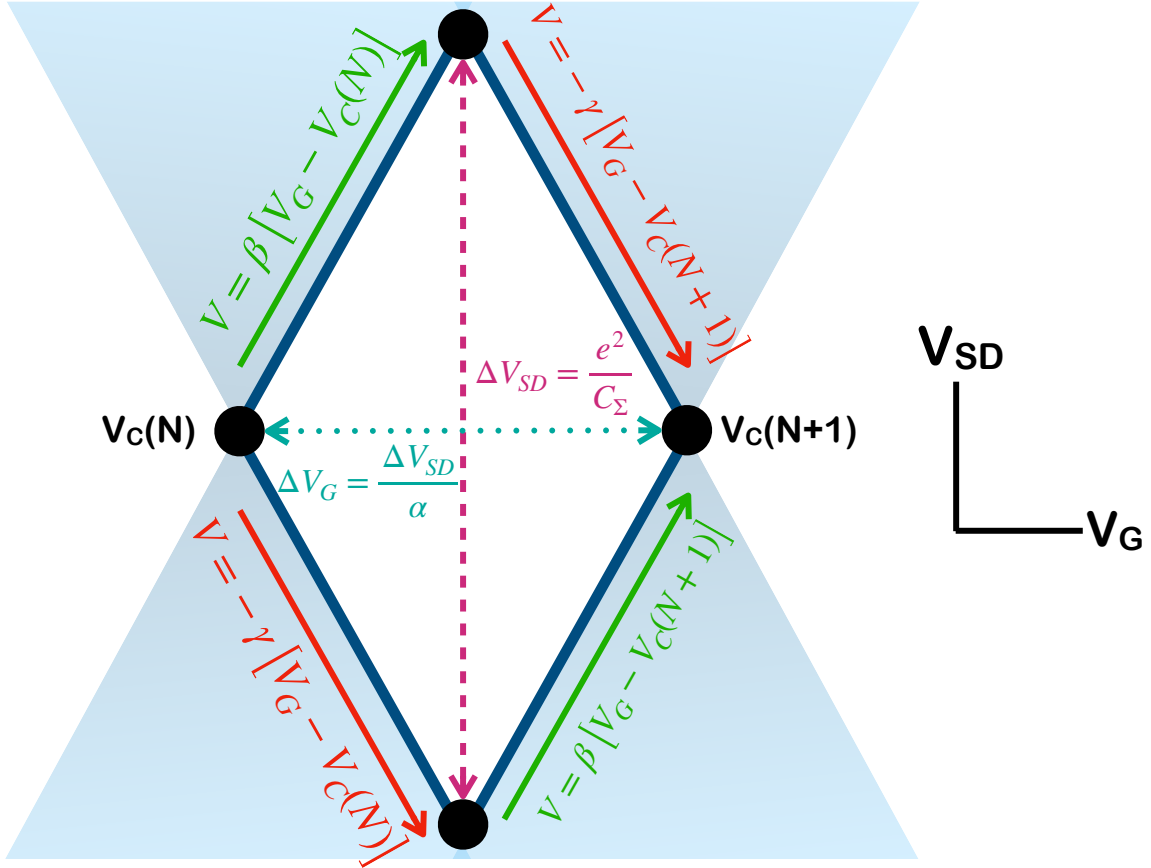


Figure 2.7: Schematic of a Coulomb diamond obtained from a bias spectroscopy measurement of a single quantum dot [51]. Areas outside the diamond exhibit current and inside the diamond is Coulomb blockade.

dot,

$$\diamond_{\text{height}} = \frac{e^2}{C_\Sigma}, \quad (2.7)$$

$$\diamond_{\text{width}} = \frac{\diamond_{\text{height}}}{\alpha}, \quad (2.8)$$

where α is known as the gate coupling factor or more commonly as the lever arm of the quantum dot (units of eV/V). This is a measure of the conversion factor from gate voltage to a change in electro-chemical energy levels. It can be obtained from the ratio of the

width to height or by using the two slopes mentioned earlier,

$$\alpha = \left(\frac{1}{\beta} + \frac{1}{\gamma} \right)^{-1}. \quad (2.9)$$

A parallel plate capacitor has capacitance,

$$C_{\Sigma} = \frac{\epsilon_R \epsilon_0 A}{d}, \quad (2.10)$$

where d and ϵ_R are the dielectric oxide's thickness (separating the dot and the surface gates) and relative permittivity respectively, where A represents the area of the parallel plate. By modelling the quantum dot as a parallel plate capacitor with a circular profile, we can get an idea of the lateral extent of the dot [33],

$$r \simeq \sqrt{\frac{C_{\Sigma} d}{\epsilon_R \epsilon_0 \pi}}. \quad (2.11)$$

In summary, Coulomb diamond energy spectroscopy is often used to characterize the quantum dot and obtain relevant properties about it to guide the tuning process.

Charge Sensing

As can be seen in Figure 2.8, Coulomb peaks are very sharp (at sufficiently low temperature), which means that the current through a quantum dot is very sensitive to gate voltage. This also means that it is sensitive to external electric fields. If there were any disturbances in the electric field (say by a nearby charge moving in the semiconductor), the electro-chemical potential through the dot would change, which can also strongly affect the current through the dot. We can use this to our advantage by constructing a “charge sensor” dot nearby other quantum dots we wish to study. This technique is applied to coupled quantum dot systems as a method of measuring charge rearrangements. Assuming one has a quantum dot formed in their device, it can be tuned as a charge sensor by:

1. Sweeping the voltage on the gate to display a set of Coulomb blockade peaks.
2. Obtain the transconductance of the data.
3. Fix the gate voltage at the point of highest conductance.

If done successfully, disturbances in the electric field would result in a perturbation, δV_G , (due to the capacitive network) which causes an abrupt change in current through the charge sensing dot, δI_{SD} , as shown in Figure 2.8. This is a double-edged sword; as when using charge sensors to read out nearby quantum dots, unintentional disturbances in the nearby electric field (e.g. due to fluctuating charges in the semiconductor or oxide material) can also cause unintentional charge sensor activation. This can cause noise in the data collected with the charge sensor.

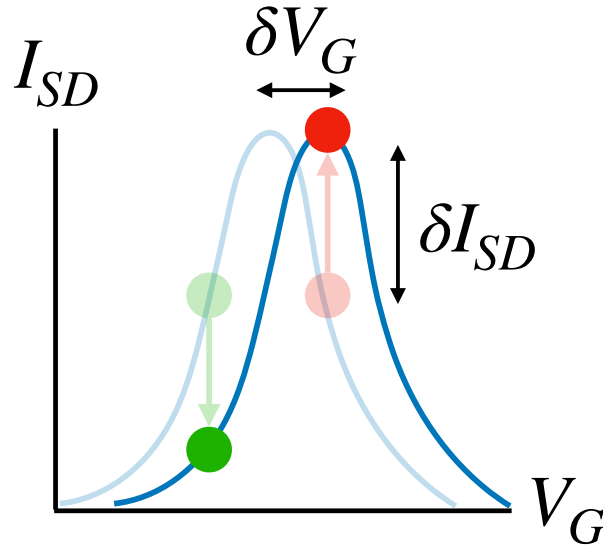


Figure 2.8: An example of a charge sensor with the gate voltage of the quantum dot fixed at a flank point (green or red). The Coulomb peak shifts due to a nearby disturbance in the electric field causing a noticeable change in current that can be recorded, indicating a perturbation.

2.3.4 Coupled Quantum Dots

Constant Interaction Model

We begin by extending the constant interaction model to multiple quantum dots. Again, this model is built on the same assumptions of validity as the single quantum dot model and is shown in Figure 2.9. What is new here is the introduction of cross capacitance values across neighbouring quantum dots.

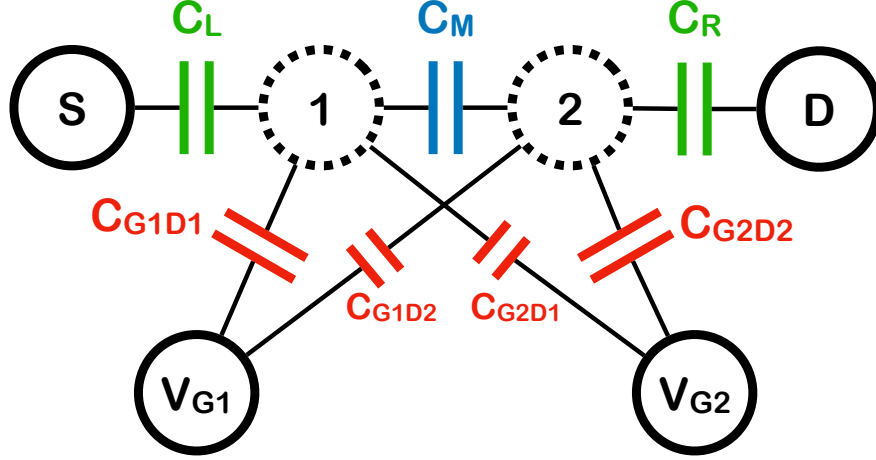


Figure 2.9: A double quantum dot modelled using the constant interaction capacitance model.

We can then write the total capacitance on each dot as a sum of the capacitance values in its parallel network [45],

$$C_{1(2)} = C_{L(R)} + C_{G1(2)D1(2)} + C_{G2(1)D1(2)} + C_M. \quad (2.12)$$

We wish to work with electro-chemical potentials,

$$\mu_1(N_1, N_2) = \left(N_1 - \frac{1}{2}\right) E_{C_1} + N_2 E_{C_M} + E_1, \quad (2.13)$$

$$\mu_2(N_1, N_2) = \left(N_2 - \frac{1}{2}\right) E_{C_2} + N_1 E_{C_M} + E_2, \quad (2.14)$$

where $E_{C_{1(2)}}$ represent the charging energies of the two dots, E_{C_M} represent the electrostatic coupling energy between the dots, and $E_{1(2)}$ are the single-particle energies for each dot.

These can be expanded to,

$$E_{C_1} = e^2 \frac{C_2}{C_1 C_2 - C_M^2}, \quad (2.15)$$

$$E_{C_2} = e^2 \frac{C_1}{C_1 C_2 - C_M^2}, \quad (2.16)$$

$$E_{C_M} = e^2 \frac{C_M}{C_1 C_2 - C_M^2}, \quad (2.17)$$

and,

$$E_1 = -\frac{V_{G1}}{|e|} (C_{G1D1} E_{C_1} + C_{G1D2} E_{C_M}) - \frac{V_{G2}}{|e|} (C_{G2D2} E_{C_M} + C_{G2D1} E_{C_1}), \quad (2.18)$$

$$E_2 = -\frac{V_{G1}}{|e|} (C_{G1D1} E_{C_M} + C_{G1D2} E_{C_2}) - \frac{V_{G2}}{|e|} (C_{G2D2} E_{C_2} + C_{G2D1} E_{C_M}). \quad (2.19)$$

We can write the electro-chemical potentials of the dots in terms of the gate voltages as,

$$\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} = \frac{e}{C_1 C_2 - C_M^2} \begin{pmatrix} C_{G1D1} C_2 + C_{G1D2} C_M & C_{G2D2} C_M + C_{G2D1} C_1 \\ C_{G1D1} C_M + C_{G1D2} C_1 & C_{G2D2} C_1 + C_{G2D1} C_M \end{pmatrix} \begin{pmatrix} \mathbf{V}_c \\ \mathbf{V}_v \end{pmatrix} \quad (2.20)$$

Charge Stability Diagram

Stability diagrams are an essential tool in the characterization of multi-dot systems. These measurement can be done in two ways: using a near-by charge sensor or measuring direct current flow through the device. Stable charge configurations of a single dot are indicated by the regions in between Coulomb peaks; crossing a peak changes the occupation by one electron. (as seen in Figure 2.6), so this diagram naturally extends to two-dimensional regions in which charge states in the two dots are stable with respect to the two gate voltages. It is at the borders of these regions where charge instability occurs and results in tunneling of charges from one dot to another. Assuming the two quantum dots are completely are uncoupled, the charge stability diagram would present itself as an orthogonal grid. In practice, the charge stability diagram will exhibit a honeycomb like pattern due to cross-capacitances and interdot capacitance. The triple points, where three different charge states become energetically degenerate and are separated in voltage space by a distance that depends on the interdot capacitance. . If the goal is to load the quantum dots into the few charge regime, the user must completely deplete the quantum dots of all

charges by locating the (0,0) charge state and then count the charge transition lines until their desired charge state is reached. This is what is known as charge tuning of the device. Figure 2.10 shows an example of charge stability diagrams for two uncoupled dots as well as two weakly coupled dots.

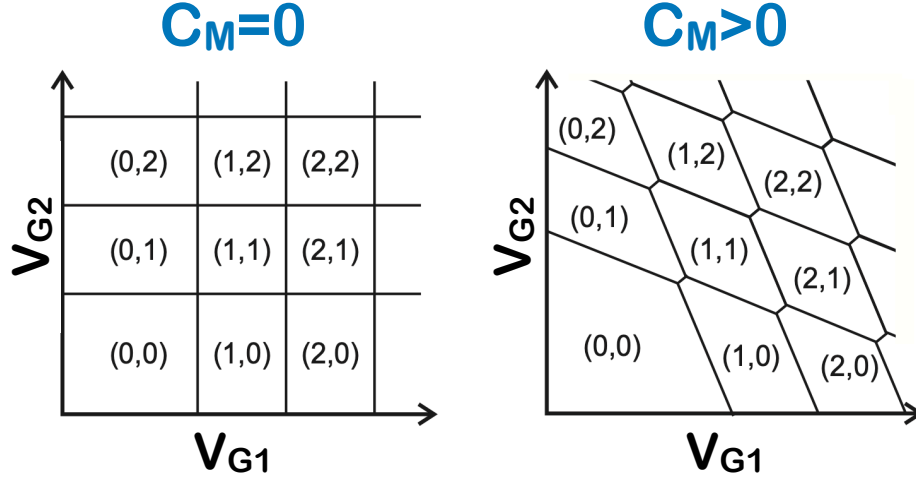


Figure 2.10: Examples of charge stability diagrams for a double quantum dot. On the left represents two uncoupled quantum dots. In practice, there will be finite capacitive coupling between the dots leading to the more realistic charge stability diagram on the right. Adapted from Hanson [17] with permission.

When the tunnel coupling between the two dots becomes significant (dots begin to merge), the charge transition lines near the triple points begin to deform [17]. Carried further, this results in the merger of the two dots into a large central one. See Figure 2.11 for an example. This is important as when one is trying to form a double dot in a device, it is beneficial to locate the voltage configuration that realizes a large central single dot and then gradually tune the tunnel coupling in order to isolate it into two dots.

Virtual Gating

Due to the cross capacitive network, we cannot change the electro-chemical potential of one of the dots without inadvertently affecting the neighbouring dots. Fortunately, we can employ the concept of *virtual gating* to achieve independent control over each dot [54, 45].

There are two types of elements in this capacitance model, the charge and the voltage nodes indicated by the c and v subscript respectively. The charge on the j th node can be

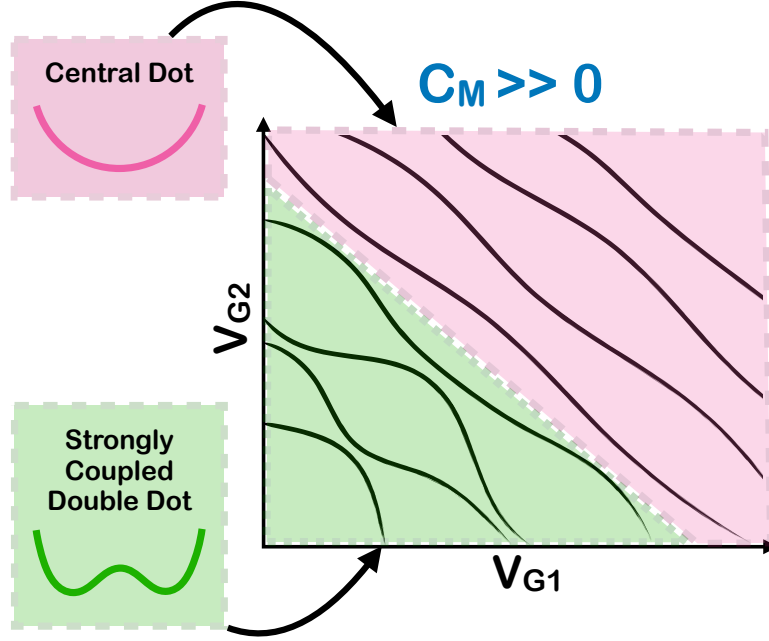


Figure 2.11: Example of the features in a charge stability diagram demonstrating how strongly coupled quantum dots can merge together.

written as the sum of all charges on the capacitors that it is connected to,

$$Q_j = \sum_{k=1}^N q_{jk} = \sum_{k=1}^N C_{jk}(V_j - V_k). \quad (2.21)$$

If we define vectors to help, we can write this as a matrix equation,

$$\mathbf{Q} = C\mathbf{V}, \quad (2.22)$$

where C is the so-called *capacitance* matrix of the quantum dot system. We can then subdivide the system into the two different kinds of nodes and re-write the matrix equation as,

$$\begin{pmatrix} \mathbf{Q}_c \\ \mathbf{Q}_v \end{pmatrix} = \begin{pmatrix} C_{cc} & C_{cv} \\ C_{vc} & C_{vv} \end{pmatrix} \begin{pmatrix} \mathbf{V}_c \\ \mathbf{V}_v \end{pmatrix}, \quad (2.23)$$

Each of these sub-matrices represent something physical about the system. Off-diagonal elements of C_{cc} represent the capacitance between charge nodes while off diagonal elements

of C_{vv} represent the capacitance between voltage nodes. Elements of C_{cv} are useful as they give the capacitances between the voltage and charge nodes.

For simplicity, consider the case of the double quantum dot,

$$C_{cc} = \begin{pmatrix} C_1 & -C_M \\ -C_M & C_2 \end{pmatrix}, \quad (2.24)$$

$$C_{cv} = \begin{pmatrix} -C_{G1D1} & -C_{G2D1} & -C_L & -C_{RD1} \\ -C_{G1D2} & -C_{G2D2} & -C_{LD2} & -C_R \end{pmatrix}, \quad (2.25)$$

where we have included the source and the drain in the capacitance matrix. This allows us to write a set of virtual gate voltages $\mathbf{U} = G\mathbf{V}$, where G is the matrix describing the voltage conversion between the two basis sets,

$$\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} = |e| C_{cc}^{-1} C_{cv} \begin{pmatrix} \mathbf{V}_c \\ \mathbf{V}_v \end{pmatrix}, \quad (2.26)$$

where it is clear that $G = C_{cc}^{-1} C_{cv}$. Now the question is how do we extract these capacitance matrix terms? The answer is by using the charge stability diagram of pairwise dots as shown in Figure 2.12 [45].

$$C_{G1(2)D1(2)} = \frac{\Delta V_{G2(1)}}{\Delta V_{G1} \Delta V_{G2} - \Delta V_{G1}^c \Delta V_{G2}^c} |e|, \quad (2.27)$$

$$C_{G1(2)D2(1)} = \frac{\Delta V_{G2(1)}^c}{\Delta V_{G1} \Delta V_{G2} - \Delta V_{G1}^c \Delta V_{G2}^c} |e|, \quad (2.28)$$

$$\frac{C_{(1,2)}}{C_M} = \frac{|e|}{C_{G2(1)D2(1)} \Delta V_{G2(1)}^M} - \frac{C_{G2(1)D1(2)}}{C_{G2(1)D2(1)}}, \quad (2.29)$$

and with finite source-drain bias voltage we obtain,

$$|e| \delta V_{G1(2)} = \frac{C_1 C_2 - C_M^2}{C_{G1(2)D1(2)} C_{2(1)} + C_{G1(2)D2(1)} C_M} |e V_{SD}|. \quad (2.30)$$

The power of virtual gating is that we can have independent control over each dot. This is becoming a commonly used protocol in the tuning up of multi-dot systems as spin qubit registers. For example, virtual gating was used in the demonstration of high-fidelity single charge shuttling across a linear array of 9 quantum dots 2.13.

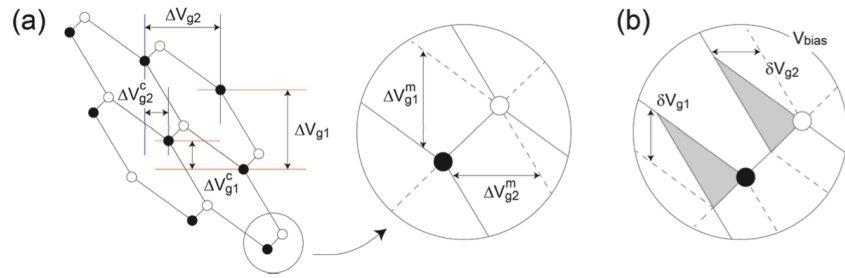


Figure 2.12: (a) Schematic of a charge stability diagram showing the gate voltage differences that should be used for calculating the capacitance of the network. (b) The presence of bias triangles due to a finite source drain bias across the device. Adapted from Penfold-Fitch [45] with permission.

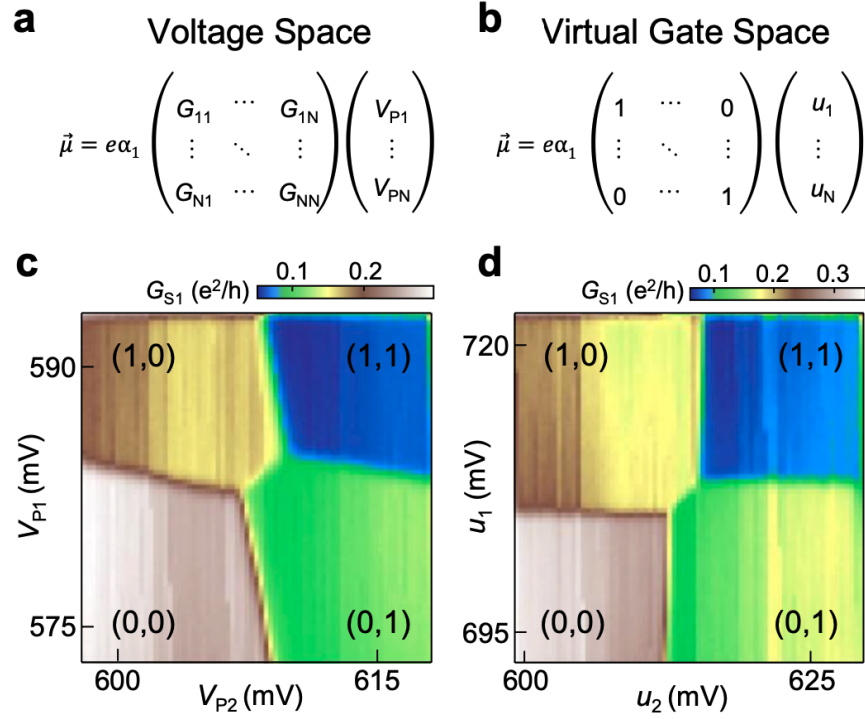


Figure 2.13: (c,d) Double quantum dot charge stability diagrams before and after virtual gating. One can see that the charge stability diagram has been orthogonalized using the virtual gates (a,b) indicating one can isolate and address specific electro-chemical potentials without impacting nearby dots [36].

2.4 Quantum System Dynamics

It is known that an *isolated* quantum system will evolve according to the Schrödinger equation. In reality, the system interacts with the surrounding environment causing non-unitary evolution and decoherence. Analogous to classical mechanics, Newton's laws are sufficient for describing the evolution of an isolated object. Once you extend the system to the real world where there are dissipating forces, you need to update the equation of motion (in this case add terms modelling the interaction). The goal of the following section is to guide the reader towards a master equation that describes the evolution of a quantum state subject to environmental interactions.

2.4.1 Open Quantum System Dynamics

The equation of motion for a density matrix is famously given by the Liouville-Von Neumann equation [5],

$$\frac{d}{dt}\rho(t) = -i[H(t), \rho(t)] \quad (2.31)$$

However, this equation is only good for a system that is assumed to be closed and isolated from its environment. The goal is to extend this equation under the assumption that the system under consideration is embedded in an environment that it is allowed to interact with.

An open quantum system is a system S that is coupled to another quantum system B known as the *environment*. We often consider the combined system $S + B$ as being closed. The state of the subsystem S will be influenced by its own internal dynamics as well as its interaction with the surroundings. This inter-system interaction results in the subsystem S no longer being able to be represented in terms of standard unitary Hamiltonian dynamics. In order to formulate the theory behind this open system, we denote $\mathcal{H}_S$ and $\mathcal{H}_B$ as the Hilbert spaces for the system and environment respectively. The Hilbert space for the total system $S + B$ can be given by the tensor product space $\mathcal{H} = \mathcal{H}_S \otimes \mathcal{H}_B$. Now, the total Hamiltonian of the system $H(t)$ is of the form,

$$H(t) = H_S \otimes \mathbb{I}_B + \mathbb{I}_S \otimes H_B + \hat{H}_I(t), \quad (2.32)$$

where H_S is the self-Hamiltonian of the open system S , H_B is the free Hamiltonian of the environment B , and $\hat{H}_I(t)$ is the Hamiltonian that describes the interaction between system and the environment. The observables referring to S are all of the form $A \otimes \mathbb{I}_B$ where A is an operator acting on $\mathcal{H}_S$ and $\mathbb{I}_B$ is the identity operator in $\mathcal{H}_B$. If $S + B$ is

described by a density matrix ρ , then the expectation value for all observables on S are governed by,

$$\langle A \rangle = \text{Tr}_S \{ A \rho_S \}, \quad (2.33)$$

where $\rho_S = \text{Tr}_B \rho$ is the reduced density matrix pertaining to the system S . The total density matrix evolves unitarily so we can write,

$$\rho_S(t) = \text{Tr}_B \{ U(t, t_0) \rho(t_0) U^\dagger(t, t_0) \}, \quad (2.34)$$

where $U(t, t_0)$ is the time-evolution operator for the total system. The equation for the reduced density matrix is obtained by tracing out the environmental degrees of freedom,

$$\frac{d}{dt} \rho_S(t) = -i \text{Tr}_B [H(t), \rho(t)]. \quad (2.35)$$

Now it is important to note that this is the *exact* dynamical equation for the system under consideration. Physical open quantum systems lack a complete mathematical model of the combined system dynamics; this results in many approximations having to be made to obtain an analytical equation that can be solved.

2.4.2 Dynamical Maps

In order to understand the dynamics of the reduced system under study we need to introduce the idea of a dynamical map. We begin by supposing the state of the total system $S+B$ is prepared at an initial time $t = 0$ as an uncorrelated product state $\rho(0) = \rho_S(0) \otimes \rho_B$ where $\rho_S(0)$ is the initial state of the reduced system and ρ_B is the reference state of the environment. We can write the evolution of the reduced system to some time $t > 0$ as,

$$\rho_S(0) \mapsto \rho_S(t) = V(t) \rho_S(0) \equiv \text{Tr}_B \{ U(t, 0) [\rho_S(0) \otimes \rho_B] U^\dagger(t, 0) \}. \quad (2.36)$$

Regarding the environment and the final time to be fixed, this relation defines a mapping from the space of density matrices for the reduced system into itself,

$$V(t) : \mathcal{S}(\mathcal{H}_S) \rightarrow \mathcal{S}(\mathcal{H}_S). \quad (2.37)$$

This map fully describes the state change of the open system over a time t and is known as a *dynamical map* [5]. What's crucial is that it is completely positive and trace preserving (quantum information is not lost). If we now allow t to vary we end up with a one-parameter family of dynamical maps: $\{V(t) | t \geq 0\}$ with $V(0) = \mathbb{I}$ (which form the quantum dynamical semigroup).

2.4.3 Markovian Processes

Interactions with the environment can either be Markovian or Non-Markovian. Markovian dynamics are when the system's evolution depends only on its current state and not on any past history; so we say that the system is 'memoryless'. Non-Markovian dynamics are when memory effects play a role and the evolution depends on its entire history. The study of non-markovian dynamics is rather challenging and remains an activate area of research. Moving forward, the Markovian approximation will be assumed; which has semigroup property,

$$V(t_1)V(t_2) = V(t_1 + t_2), \quad t_1, t_2 \geq 0. \quad (2.38)$$

2.4.4 Master Equation

Given a quantum dynamical semigroup that exhibits Markovian processes, there exists a linear map $\mathcal{L}$ which is the generator of the semigroup. We can represent it in exponential form as,

$$V(t) = \exp(\mathcal{L}t), \quad (2.39)$$

which yields a first-order differential equation for the reduced density matrix of the open system,

$$\frac{d}{dt}\rho_S(t) = \mathcal{L}[\rho_S(t)], \quad (2.40)$$

which is known as the Markovian quantum master equation. In order to construct the most general form of the generator, we consider a finite dimensional Hilbert space $\mathcal{H}_S$ where $\dim \mathcal{H}_S = N$. The corresponding Liouville space is a complex space of dimension N^2 . It can be derived that the diagonal form of the generator is obtained as,

$$\frac{d}{dt}\rho_S(t) = \mathcal{L}[\rho_S(t)] = -i[H_S(t), \rho_S(t)] + \sum_{k=1}^{N^2-1} \gamma_k \left(L_k \rho_S(t) L_k^\dagger - \frac{1}{2} \left\{ L_k^\dagger L_k, \rho_S(t) \right\} \right), \quad (2.41)$$

where $\{A, B\} = AB + BA$ is the anti-commutator [5]. The first term represents the unitary part of the dynamics generated by the Hamiltonian H , the operators L_k are the *Lindblad* or *quantum jump* operators, γ_k play a role of relaxation rates for the different decay modes of the open system. It is important to note that these Lindblad operators are often defined phenomenologically or can be derived from first principles [38].

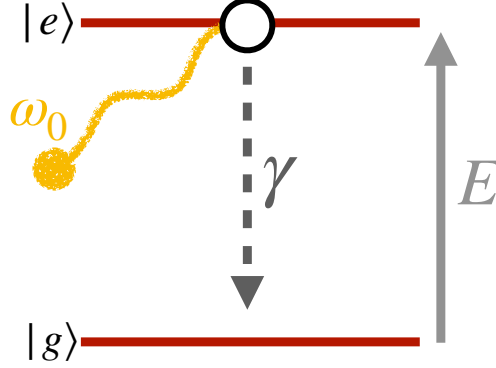


Figure 2.14: Schematic of a two level open quantum system the can spontaneously emit energy and transition energy states through spontaneous relaxation or excitation.

Two-Level System Example

Consider a two-level system shown in Figure 2.14. We know that the system Hamiltonian of this two-level system is,

$$H_S = \frac{1}{2}\omega_0\sigma_z \quad (2.42)$$

with Hilbert space $\mathcal{H} = \text{span}\{|e\rangle, |g\rangle\}$. Phenomenologically, we say that the excited states can relax at a transition rate γ , with the $L = \sigma_-$ being the jump operator for that process. This results in the following master equation,

$$\frac{d}{dt}\rho_S(t) = -i\frac{\omega_0}{2}[\sigma_z, \rho_S(t)] + \gamma \left(\sigma_- \rho_S(t) \sigma_+ - \frac{1}{2}\{\sigma_+ \sigma_-, \rho_S(t)\} \right) \quad (2.43)$$

This equation can then be solved using any differential equation solver to obtain the evolution of the initial density matrix.

2.5 Quantum Information Processing

2.5.1 DiVincenzo's Criteria

With an understanding of quantum dot semiconductor devices and open quantum systems, we can now discuss their utility as a computing platform. We must begin with what makes

a system suitable for quantum computing; a set of five criterion initially posed by David P. DiVincenzo [9],

1. A scalable physical system with well characterized qubits.
2. The ability to initialize the state of qubits into a simple fiducial state.
3. Long relevant decoherence times, much longer than the gate operation time.
4. A 'universal' set of quantum gates.
5. A qubit-specific measurement capability.

2.5.2 The Qubit

In order to form a qubit we need a two-level system defined by a characteristic energy gap. There are many ways of defining qubits in semiconductor quantum dots but the assumed in this thesis is using the spin projections of a single electron as the logical states [34]. The energy splitting can be tuned by an external applied magnetic field (Zeeman effect). There are many trade-offs using this encoding, as although it has better coherence times, spin manipulation times are much slower.

2.5.3 State Initialization

Initial state preparation can be realized by applying a uniform magnetic field and cooling the system [34]. This achieves the desired initial spin polarization prior to gate operation.

2.5.4 Sources of Decoherence and Gate Times

There are two dominant sources of decoherence that affect a spin states evolution. They are characterized by two time constants, T_1 and T_2 , known as spin relaxation and dephasing respectively [34].

Spin Relaxation

This time constant occurs from spin interaction with the environment around it, causing it to lose energy through energy dissipation. This results in the qubit only being able to stay in an excited state for a certain amount of time until it fundamentally “relaxes” to a lower energy state. A longer T_1 time is desirable as it ensures the qubits spin state is coherent through out the quantum operation. Standard T_1 times in silicon semiconductor devices are on the order of tens of *milliseconds*.

Spin Dephasing

This time constant also occurs due to spin interactions with its environment but it affects the phase coherence. The source of such error can vary; from spin-spin interactions with the environment to charge noise due to fluctuating charge traps near the spin. Typically this time constant is much shorter than T_1 and so poses a greater threat to the coherence of the spin qubit. Standard T_2 times in silicon quantum dots are around tens of *microseconds*.

Gate Operation Times

Typical gate operations are designed to be less than a microsecond, with shorter being better. It is imperative to keep the pulses much less than the characteristics T_2 time to prevent any decoherence errors.

2.5.5 Universal Set of Quantum Gates

Single-Qubit Operations

The standard method for single-qubit spin manipulation is through electron spin resonance (ESR) [34]. The Hamiltonian for the interaction of a spin with an external magnetic field is given by,

$$H = g\mathbf{B} \cdot \mathbf{S} = \frac{1}{2}g\mu_B\mathbf{B} \cdot \boldsymbol{\sigma}, \quad (2.44)$$

So if we consider a static Zeeman field $\mathbf{B}_0$ applied onto the system, we obtain the evolution operator,

$$U = \exp\left(-i\frac{t}{2\hbar}\mathbf{B}_0 \cdot \boldsymbol{\sigma}\right), \quad (2.45)$$

which describes a constant spin precession around the axis of the field with the Larmor frequency $\omega_0 = \frac{g\mu_B B_0}{\hbar}$. This only achieves rotation around the axis of the applied static field (typically defined as the z axis). To obtain rotation around another axis, we need to apply an oscillating radio frequency magnetic field $\mathbf{B}_{RF} = B_{RF}(\cos \phi \mathbf{x} + \sin \phi \mathbf{y})$ in a perpendicular direction. The Hamiltonian of the combined field read as,

$$H = \frac{\hbar\omega_0}{2}\mathbf{Z} + \frac{\Omega}{2}[\mathbf{X}\cos(\omega_{RF}t + \phi) + \mathbf{Y}\sin(\omega_{RF}t + \phi)], \quad \Omega := g\mu_B B_{RF}, \quad (2.46)$$

Moving to a reference frame that rotates with the Larmor precession,

$$|\psi_{\text{rot}}\rangle = \exp\left(i\frac{\omega_0}{2}\mathbf{Z},\right)|\psi_{\text{lab}}\rangle \quad (2.47)$$

We get the Hamiltonian,

$$H_{\text{rot}} = (\omega_0 - \omega_{RF})\frac{\mathbf{Z}}{2} + \frac{\Omega}{2}(\mathbf{X}\cos \phi + \mathbf{Y}\sin \phi). \quad (2.48)$$

If the frequency of the oscillating field is tuned to be in resonance with Larmor frequency ($\omega_0 = \omega_{RF}$), the qubit will realize rotation around the radio-frequency field. Conversely, one can disable the radio-frequency field by zeroing its amplitude to obtain rotations purely around the Zeeman field axis. Therefore, any single-qubit operation can be realized simply by tuning the parameters ω_{RF} , B_{RF} and ϕ . We will see later that tuning ω_{RF} can be avoided by directly modulating the g -factor of each spin qubit independently through external voltages.

Two-Qubit Operations

The key component to a two qubit operation is the exchange coupling between two qubits. In this case, the exchange interaction between two particles arise from the symmetrization requirement of identical fermions. This results in the triplet spin configuration to be associated with a higher energy orbital state; captured through the Heisenberg interaction,

$$H = \frac{J}{4}\vec{\sigma}_1 \cdot \vec{\sigma}_2, \quad (2.49)$$

where the constant J represents the strength of this interaction [34]. This term arises due to the overlap between the electron wavefunctions and so can be tuned by changing the tunnel coupling between the two electron spins. This was first proposed by Loss and DiVincenzo where their idea was to effectively turn off and on the tunnel coupling between

spins to control the duration of their interaction [34]. Since singlet and triplet states are eigenstates of this Hamiltonian, they will oscillate back and forth under this effect allowing one to deterministically exchange the spins.

The SWAP gate, as the name suggests, comes from evolving two spin states under the Heisenberg exchange interaction for a specific duration that “swaps” the spins. For example, evolving the state $|\uparrow\downarrow\rangle$ to $|\downarrow\uparrow\rangle$. This gate combined with single qubit rotations from ESR *do not* form a complete set of quantum operations. For this, we need what is called the $\sqrt{\text{SWAP}}$; evolving $|\uparrow\downarrow\rangle$ to the triplet configuration $\frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle + |\downarrow\uparrow\rangle)$ [25].

Together, single qubit rotations and the $\sqrt{\text{SWAP}}$ gate can be combined to realize *any* spin evolution!

2.5.6 Measurement

Measuring a single magnetic spin is difficult and so in semiconductor quantum dots, the primary method to measure the spin state of a qubit is through some implementation of spin-to-charge conversion [18]. This allows the spin state of an electron to be mapped directly to a binary current measurement. Measurement read-out is beyond the scope of this thesis but the reader is suggested to consult the following sources for review [7, 6].

2.6 Summary

In this section, we learned about quantum dot semiconductor devices and how to model their behaviour to give context to fundamental device measurements. This was followed with the introduction to quantum systems and how the system density matrices evolve in an external environment. Finally, we learned about semiconductor spin qubits as a platform for quantum computing, showcasing how they pass all of DiVincenzo’s criteria.

Chapter 3

Machine Learning

3.1 Deep Learning

Deep learning is a tool that allows computational models, which are composed as multiple layers of processing nodes, to learn abstract representations of data [30]. The architecture of deep learning shows up in all facets of machine learning; any algorithm that requires tuning a set of parameters to optimize an objective will utilize principles of deep learning.

3.1.1 Automatic Differentiation

A parameter network in deep learning is the computational network that connects the model's parameters to an output through a series of functions. The output is known as an objective (or cost) function; a metric that measures the success of your model. In order to understand how to tune the model's input in a way that improves the objective, we need to understand how the output varies with the input - the gradient!

This is done by propagating the product of locally calculated gradients backwards through the parameter network, otherwise known as *back-propagation*. There are many ways to calculate gradients on a computer with each method having trade-offs. Manual differentiation is time consuming and prone to sources of error. Numerical differentiation can be inaccurate due to round-off errors and scales poorly. Symbolic differentiation results in complex expressions that require models to be defined as closed-form expressions. For this reason, automatic differentiation was considered as it computes and accumulates local gradients as the parameter network is traversed *during* the model execution [3]. This allows

for the calculation of gradients at the machine's precision while only have a small constant factor of overhead [3]. Consequently, the gradient of the objective function with respect to the parameters can be obtained by traversing the network backwards and exploiting the chain rule. Figure 3.1 demonstrates both a forward- and backward-propagation pass through a simple parameter network.

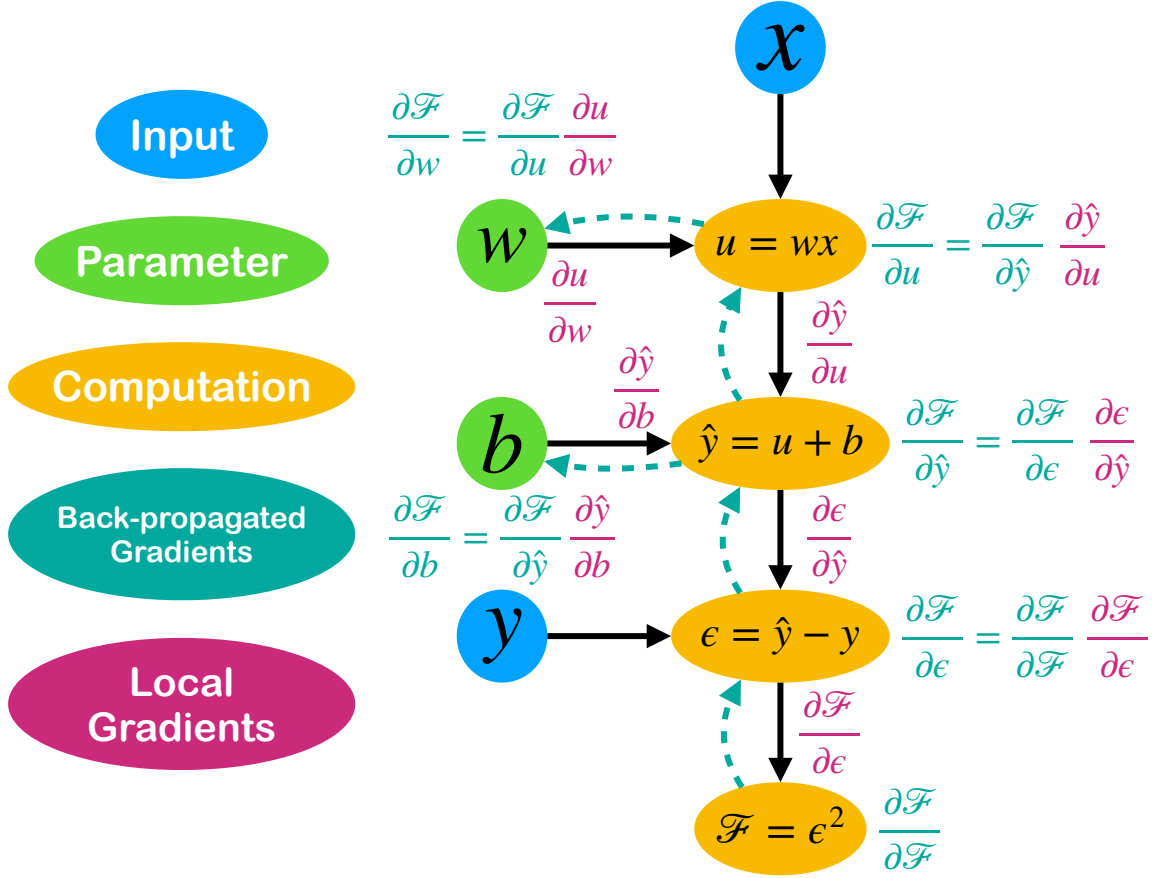


Figure 3.1: Example of a parameter network with inputs x , output y and ideal output $\hat{y}$. The goal is to find parameters, w and b , such that the objective function $\mathcal{F} = (y - \hat{y})^2$ is minimized. Local gradients (in purple) are calculated during the forward pass of the network and then by traversing the network backwards and computing the product of local gradients; parameter sensitivities (blue) can be found.

3.1.2 Parameter Optimization

The goal of parameter optimization is to efficiently tune the parameters in a way that *improves* the objective function. For this we can use the standard gradient descent protocol to update the control parameters. There are a variety of gradient descent update rules that can be used but the most popular is the Adam optimizer; being an extension of Stochastic Gradient Descent (SGD) combining the advantages obtained from AdaGrad and RMSProp [26]. Adam is an adaptive gradient descent algorithm, meaning it keeps track of prior gradient history to maintain a constant learning rate per model parameter. This optimizer is a suitable choice for many problems as it has a lot of advantages,

- Default hyper-parameter values do well on a wide variety of problems.
- First-order gradient-based optimization that is computationally efficient and has little memory requirements.
- Well suited for problems that are larger in terms of data and/or parameters.

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t .

Require: α : Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0 : Initial parameter vector
 $m_0 \leftarrow 0$ (Initialize 1st moment vector)
 $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
 $t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
 $t \leftarrow t + 1$
 $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
 $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
 $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
 $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
 $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
 $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

Figure 3.2: Algorithmic parameter update rule used by the Adam optimizer [26]. Adam employs many adjustments to the gradient to help improve optimization convergence and accuracy.

From Figure 3.2, a vector of model parameters $\vec{\Theta}$ should be updated every n^{th} iteration according to,

$$\vec{\Theta}^{(n)} \leftarrow \vec{\Theta}^{(n)} - \eta^{(n)} \frac{\hat{m}^{(n)}}{\sqrt{\hat{v}^{(n)}} + \epsilon}, \quad (3.1)$$

where $\hat{m}^{(n)}$ is calculated directly related to the gradient $\nabla_{\vec{\Theta}} \mathcal{F}$. The quantity $\eta^{(n)}$ represents the learning rate and is a direct analogue of how large of a “step” the optimizer takes towards the gradient. This is an example of a “hyper-parameter” that directly impacts the optimization performance. These hyper-parameters are often tuned heuristically by the user which could lead to sub-optimal model performance. If the learning rate is too small then the model will take forever to converge. If the learning rate is too large, the model might be unable to converge to a solution. Figure 3.3 shows a two-parameter optimization problem with a complex objective function that is plagued with local minima.

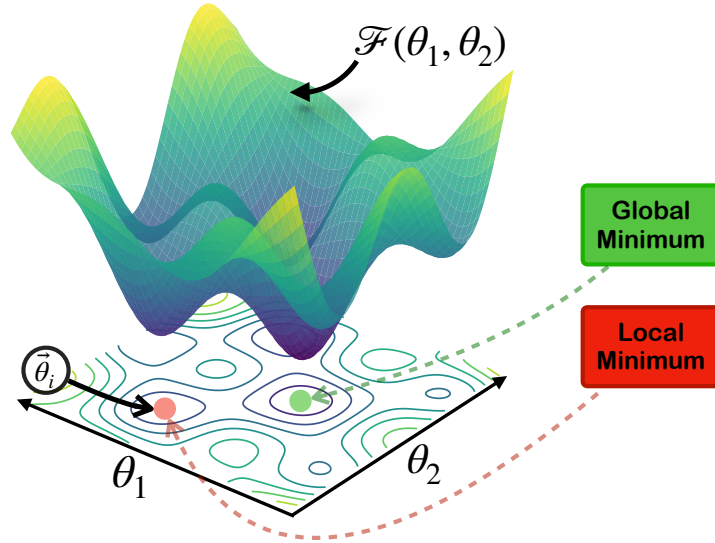


Figure 3.3: Gradient descent parameter optimization applied to a two parameter, highly non-convex, objective function $\mathcal{F}$. From the initial parameters $\vec{\theta}_i$, the optimizer follows the gradient (black arrow) towards the sub-optimal local minimum. Since the model strictly follows the gradient, it will remain at the local minima perpetually.

For problems with non-convex objective functions, gradient-descent based techniques often fail to converge to the optimal solution; getting trapped in local minima. This can be

mitigated with the introduction of stochastic processes into the update rule to help explore the parameter space more effectively. Adapting the learning rate during the optimization process to hopefully “step over” the local minima on the way to the optimal solution is a common technique as well.

3.1.3 Langevin Dynamics

There are two things we can do to help prevent getting trapped in local minima: learning rate adjustments and adding stochastic processes to the parameter updates. The former is simple and just involves updating the learning rate according to some schedule. The latter can be done by appending a noise term to our update rule where the presence of the term depends on the “temperature” of the system [12]. When the system is “hot,” exploration of the parameter space is prioritized by introducing significant random perturbations. As the system “cools”, the gradient becomes more dominant, focusing on exploiting the gradient of the objective function. A similar approach was successfully applied to a three-qubit NMR system showing faster convergence when compared to just using gradient descent alone [47].

Assuming a set of model parameters $\vec{\Theta}$, the temperature, learning rate and parameters are updated with,

$$T^{(n)} \leftarrow f(T^{(n)}, \eta^{(n)}, n) \quad (3.2)$$

$$\eta^{(n)} \leftarrow g(\eta^{(n)}, n) \quad (3.3)$$

$$\vec{\Theta}^{(n)} \leftarrow \vec{\Theta}^{(n)} - \eta^{(n)} \frac{\hat{m}^{(n)}}{\sqrt{\hat{v}^{(n)}} + \epsilon} + \sqrt{2T^{(n)}} \delta \vec{\Theta} \quad (3.4)$$

where f and g are arbitrary scheduling functions, $T^{(n)}$ is the temperature at that iteration, and $\delta \vec{\Theta} \sim \mathcal{N}(0, 1)^{\dim \vec{\Theta}}$ is a vector sampled from the standard normal distribution. The optimal trade off between exploration and exploitation is demonstrated in Figure 3.4. Figure 3.5 shows an example of a possible learning rate and temperature schedule used in this thesis.

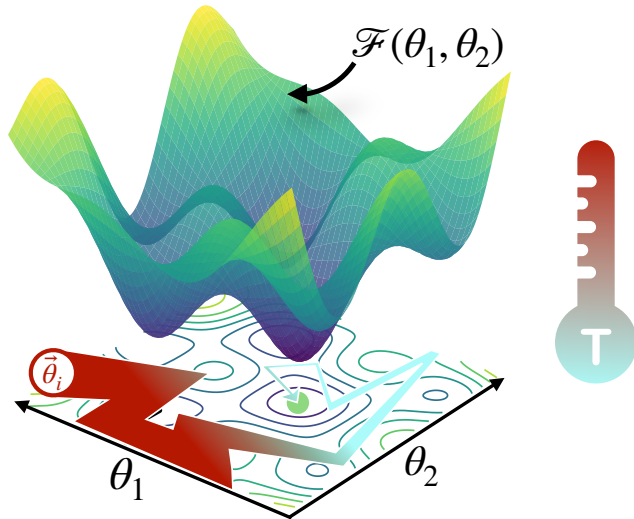


Figure 3.4: Langevin dynamics optimization applied to a two parameter, highly non-convex loss landscape. As the temperature of the system is initially hot, the optimization trajectory jumps around to explore the landscape. As it begins to cool, it exploits the gradient and is able to find the global minima.

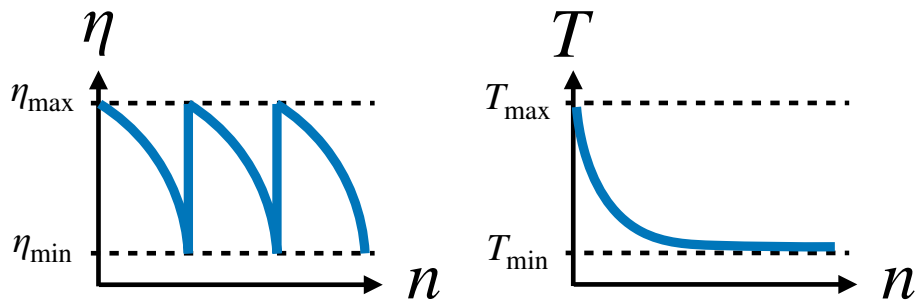


Figure 3.5: (left) Example of a possible learning rate annealing schedule. It follows a cosine annealing schedule with warm restarts. This is beneficial as having a variable step size for gradient descent can help to speed up convergence and/or escape local minima. (right) Example of a temperature schedule that monotonically decreases with epoch number (for example $f(n) = T_{\max}/n + T_{\min}$). This prioritizes parameter exploration while the system is hot and switches to exploiting gradients when the system cools down.

3.2 Computer Vision Techniques

3.2.1 Edge Detection

The Hough transform is a technique used to detect lines and curves in pictures [10]. The essence of this technique is that a set of points in image space can be mapped to a point in parameter space as shown in Figure 3.6. If the image points lie on the same line, the curves will intersect. The algorithm detects these intersections and processes them to obtain the true lines present in the image. However, lines are often unintentionally merged together and/or image noise can cause lines to be missed. This is why the image usually needs to be pre- and post-processed strategically to assist the Hough transform. Figure 3.7 shows an example of a Hough transform applied to an image.

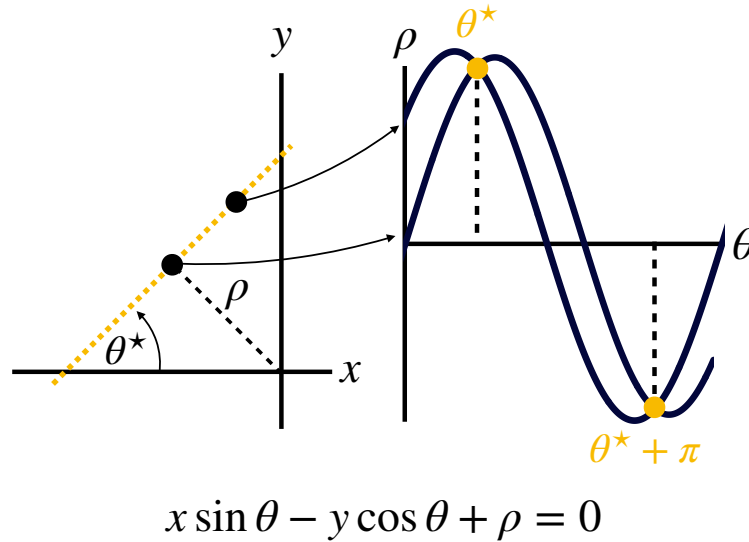


Figure 3.6: A demonstration of how points in image space (left) are mapped to Sinusoidal curves in parameter space (right). The intersection of the sinusoidal curves represent lines in image space. The number of intersections corresponds to how likely those points correspond to a line in the image.

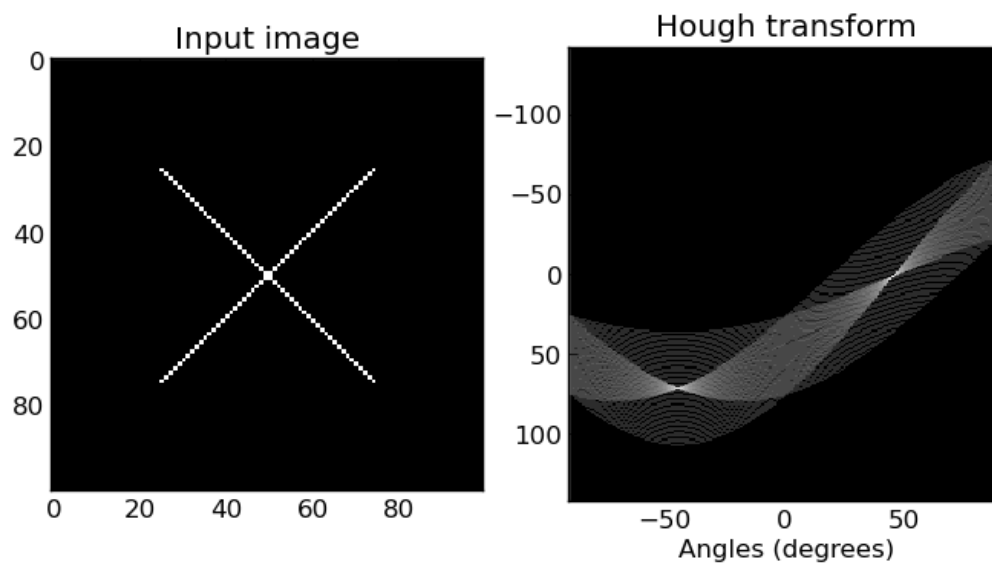


Figure 3.7: Hough transform applied to an image with two clear image lines [50]. Each point in the image (left) corresponds to a Sinusoidal curve (right) and the two clear intersection points correspond directly to the two lines seen on the image. Since the two lines are the same just rotated by ninety degrees, it explains why the two intersection points are separated by ninety degrees as well.

3.2.2 Object Detection and Classification

The field of object detection in images has continued to grow since its inception over a decade ago. Many adjacent fields, such as medicine, have benefited from the direct application of these techniques. Experimental physics is only now beginning to recognize their potential, and if it is being applied, it is usually in its most primitive form. To understand the application of the computer vision model used in this thesis, it is essential to build context on why this model was chosen over its earlier iterations.

Convolutional Neural Networks (CNN)

CNNs were the first kind of neural network tasked to perform object detection and classification [29]. This model works by convolving a kernel matrix across the image through a set of hidden layers. Through a combination of convolution and pooling layers, the final layer can be designed so that it outputs a probability vector; each component representing a different object (cat, dog, car etc). This model falls short for multiple object detection, as there may be multiple objects within the image we wish to classify. Naturally, one might think to divide the image into regions of equal size and run the model for each region, but objects are often of different size. Moreover, dividing the image into smaller regions to be run through the same CNN model would be computationally expensive. Figure 3.8 shows an example CNN model designed to predict what kind of vehicle is present in the image.

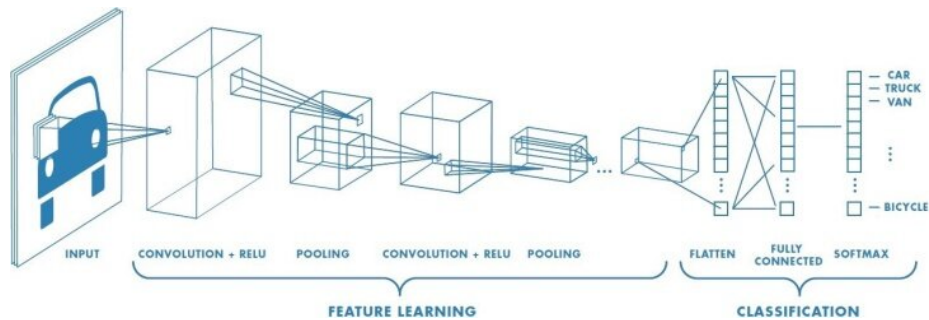


Figure 3.8: Standard CNN architecture to detect an object in an image [46]. The image data is compacted through a series of computation layers into a probability vector where each element is the probability of a certain object being represented in the image. In this case, if the model is trained correctly, the first element would be one with the rest being zero.

Region Convolutional Neural Networks (R-CNN)

R-CNNs were proposed as a fix to CNNs by being able to extract multiple regions from an image [15]. This is done through generating many candidate regions of the image and by using a greedy algorithm, the architecture recursively combines similar regions. Final region proposals are then created and sent through a CNN which extracts the features and classifies the data accordingly. In addition to the object classification, the model generates a bounding box that surrounds the object that was classified. Unfortunately, this method is very slow as it has to classify all of the proposed regions independently. Figure 3.9 shows an example model designed to locate a cowboy on his horse.

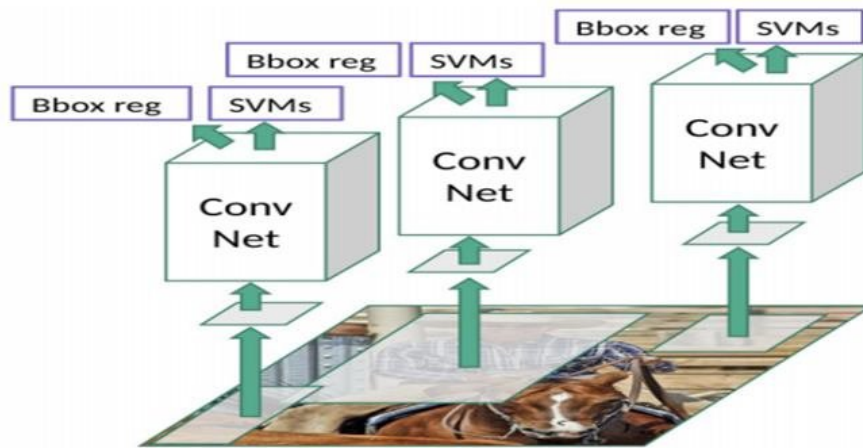


Figure 3.9: Standard R-CNN architecture to detect an object in the image [1]. The image data is compacted through a series of computation layers into a vector where each element is the probability of a certain object being represented in the image as well as bounding boxes that surround the detected object.

Fast Recurrent Convolutional Neural Networks (Fast R-CNN)

The solution to the slowness of the R-CNN model was to first pass the entire image through a CNN model, which then generates the regions of interest [14]. Next, a Region of Interest (RoI) pooling layer is used to extract a feature vector from each of these proposals. Fully connected layers are then applied to determine the class labels and bounding boxes. Figure 3.10 shows an example model designed to locate a cowboy on his horse.

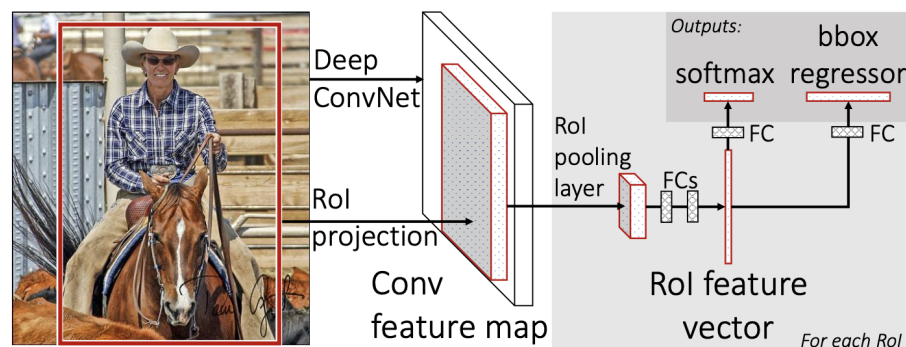


Figure 3.10: Fast-RCNN architecture designed to detect an object in the image [14]. The image data is compacted through a series of computation layers into a vector where each element is the probability of a certain object being represented in the image as well as bounding boxes that surround the detected object.

Faster Recurrent Convolutional Neural Networks (Faster R-CNN)

Faster R-CNN does not use the selective search approach and instead has a Region Proposal Network (RPN) that is trained to learn region proposals [48]. These proposals are used then by the classifier to predict the object class and subsequent bounding box. Figure 3.11 shows an example model designed to locate a cowboy on his horse.

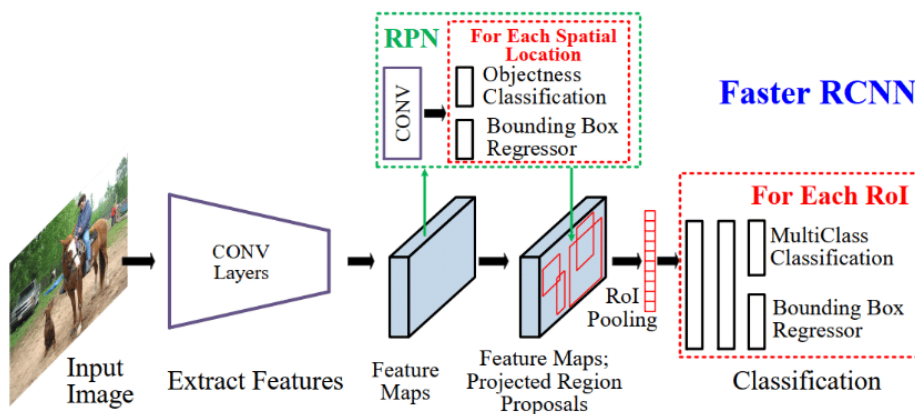


Figure 3.11: Faster-RCNN architecture to predict the object classification and generate a bounding box for what is detected [13]. It does this by first training a region proposal network and then predict each region accordingly.

Mask Recurrent Convolutional Neural Networks (Mask R-CNN)

Mask R-CNN is a general framework for object detection and segmentation from the Facebook AI Research group FAIR [19]. It is a powerful tool which can localize each detected object with a bounding box and classify each pixel into a fixed set of categories. At the time, Mask R-CNN surpassed all previous state-of-the-art computer vision models and is still being used for various image segmentation masks today [19]. The key piece of architecture in Mask R-CNN is *ROIAlign* as it is able to locate relevant objects and segment them down to the pixel level. This gives pixel level segmentation for an arbitrary number of detected objects. Figure 3.12 shows the architecture of a Mask R-CNN model applied to classifying cells in an image.

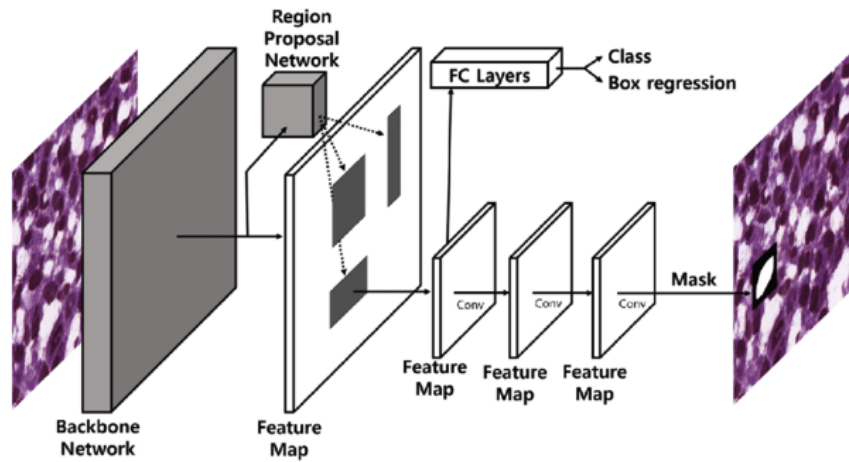


Figure 3.12: Mask-RCNN architecture designed to segment and classify certain cells in the test image [21].

Detectron2 is an open source framework built by Facebook AI Research (FAIR) and was developed using PyTorch [57]. This software allows users to easily build Mask-RCNN backbone architectures that can be used or refined by the user. This allows users to leverage the powerful abilities of the model without having to code the entire neural network by scratch. Figure 3.13 shows an example of how the model is able to segment numerous objects in an image.



Figure 3.13: An image processed using Detectron2’s Mask-RCNN model demonstrating its ability to segment and classify multiple objects in an image [57].

3.3 Summary

In the first section we learned all about stochastic parameter optimization. In it we learned about Langevin dynamics, a gradient based parameter optimization technique aided with stochastic processes to help prevent getting trapped in local minima. This is a very powerful technique can be extended to optimize control signals to generate high fidelity unitary gate operations.

In the second section we learned all about the computer vision techniques applied in this thesis. Edge detection is used for detecting lines and shapes in images. Object detection in this thesis is done through the application of Mask-RCNN; a highly refined object detection model able to segment and classify data to the pixel level.

Chapter 4

Automated Tuning of Quantum Dot Devices

4.1 Motivation

Tuning semiconductor quantum dot devices involves navigating complex experimental parameter spaces to identify optimal operating conditions. With device variability and manufacturing imperfections, tuning a device demands extensive time and resources to collect and analyze large amounts of data. My work focuses on making advancements towards achieving fully autonomous tuning, eliminating as much human intervention as possible. This work involves data analysis and computer vision techniques to extract relevant features from data to further guide the tuning process in real-time. With an automated protocol, devices can be tuned in a fraction of the time, enabling researchers to have more time to focus on exploring the interesting physics of the device.

This section covers three main software contributions: the first being a single quantum dot tuning software [44], the second is a Coulomb diamond data processing software [43] and the third is a custom image segmentation software [42]. All three pieces of software are open-source tools that any researcher can use and improve on.

4.2 Pillars of Automated Tuning

The process for initializing spin qubits in semiconductor quantum dot devices follows a standard protocol and can be divided into a sequence of five steps: bootstrapping, coarse

tuning, establishing controllability, charge state tuning, and fine tuning [62]. The end goal is to have a completely automated protocol that is able to address each of these steps efficiently and effectively. In order to provide some context, it is crucial to go over what each of those stages entail.

Bootstrapping

Bootstrapping, otherwise known as initialization, is the pre-tuning process that directly follows the cool-down of a device to cryogenic temperatures. The device is assumed to be bonded correctly and passes any fabrication error checks. The goal of this stage is to bring the device into an appropriate regime for acquiring measurement data. This means forming any desired current channels in the device and ensuring proper 2DEG formation. This stage also determines the allowed parameter variations by verifying gate response as well as determining the safe ranges for any of the gate voltages. There have been a few attempts at this with all of them using various data analysis techniques [2, 35, 8].

Coarse Tuning

Coarse tuning a device involves finding a range of gate voltages where a global configuration is established. This means defining the potential landscape in your device so that it corresponds to the desired number of dots. Ideally, at the end of this stage, a voltage configuration that realizes a desired number of dots in the current channel is realized. Computer vision was first applied by Kalantre et al. in 2019 with the application of CNNs to analyze charge stability diagrams detecting regions exhibiting double dot behaviour [22]. This method involved dividing the two-dimensional data into square chunks, which were then analyzed by the CNN to generate a probability for each dot configuration. The resulting probability space was subsequently explored to determine the voltage configuration that achieves the desired dot configuration. Darulova then improved on this by integrating a sequential script-based protocol which adjusted device gates based on the model's predicted probability vector [8]. To the best of my knowledge, no-one has advanced the complexity of the computer vision model past the CNN.

Establishing Controllability

In an ideal device, varying a single gate voltage should only impact the parameter it is designed to control (such as the electrochemical potential of a specific dot). These gates

are connected through a capacitive cross-talk network making it difficult to isolate control over a single parameter. To circumvent this problem, researchers design virtual gates, a linear combination of the system gates designed in a way such that individual dot control is achieved. The output of this phase would be a set of virtual gates for the system that can be used to establish independent control of effective system parameters. This normally doesn't require any ML techniques and can be done with standard line fitting and/or peak estimation measurements and analysis [45]. Although, a new strategy has been proposed using edge detection techniques to measure the slopes in pair-wise charge stability diagrams to extract terms in the virtual gate matrix [60].

Charge Tuning

At this stage, the device behaves as expected, with the correct number of quantum dots having been formed. The next step would be to “fill” each quantum dot in the device with the desired number of charges (typically one to three). This is commonly done by utilizing a nearby charge sensor to detect charge rearrangements in the dots of interest. The charge sensor does not provide the exact number of charges sensed but an overall direction of change. In practice, tuning the charge configuration entails emptying the dot of all charges, and then loading each dot independently by counting the appropriate transition lines. For multi-dot systems this is difficult because not only do you have to keep track of the number of charges, but also to what dot the charge belongs to. Once again, CNNs are commonly employed to tackle this issue as a way to detect and process transition lines in a charge stability diagram [11]. Their charge-tuning algorithm begins with a well-defined double QD configuration, where the initial charge state is unknown. From this point, a two-stage algorithm is used to tune the double QD to a desired charge configuration, following an unloadingreloading strategy. First the double QDs are unloaded to the (0,0) state and then reloaded to the desired (m,n) state. Each stage involves a series of measurements in the plunger gate space, analyzed by CNNs. To the best of my knowledge, no-one has advanced the complexity of the computer vision model past the CNN.

Fine Tuning

Once the system is in the desired global configuration with the correct number of charges in each quantum dot, it is ready to be prepared to use as a quantum computing device. This final stage of tuning is arguably the most difficult as it requires a lot of precision and expertise to tune the device Hamiltonian to exhibit a fully controllable multi-qubit system. Due to the scope of thesis, this stage was left for future work.

4.3 Data Segmentation Model

4.3.1 QFlow Dataset

In order to train a computer vision model to aid in tuning, a quantum dot dataset is required. Fortunately, Zwolak et al put together a public dataset called *QFlow* which consists of both simulated and experimental charge stability diagram data [63]. The dataset was designed to aid in the development and bench marking of machine learning tools. The dataset was released in two versions, QFlow 2.0 and QFlow lite; the former is the most updated version and contains both experimental and simulated data whereas the latter only has simulated data. Figure 4.1 shows a simulated charge stability diagram and its associated state labels (number of quantum dots) from the QFlow lite dataset. Although the data is being simulated based on a nanowire model, the data produced is congruent with what would be measured from a semiconductor quantum dot device.

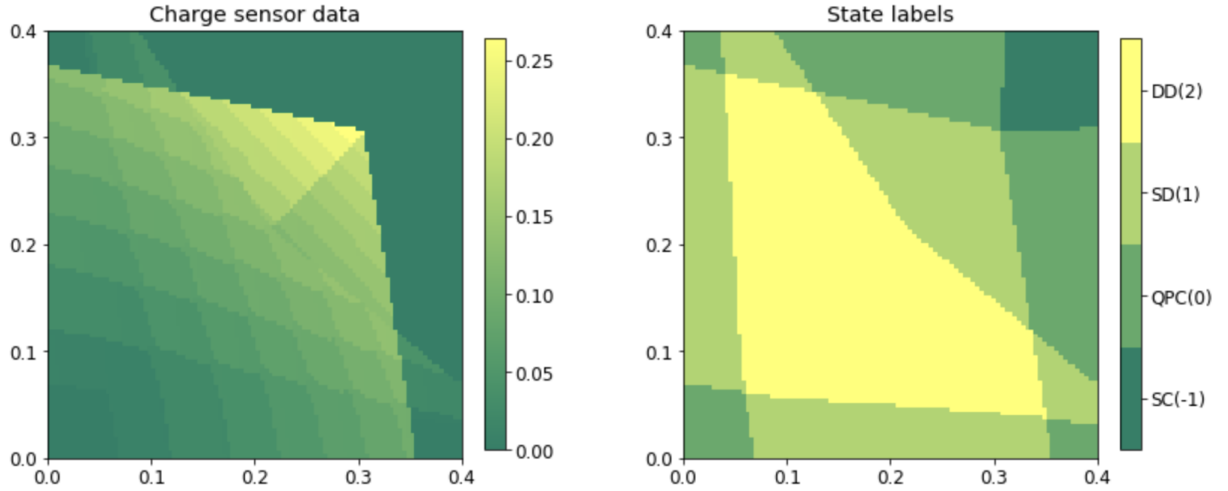


Figure 4.1: Example of a simulated image from the QFlow lite dataset [61]. The x and y axis are the plunger gate voltages defining each of the quantum dots. (left) Charge stability diagram measured from charge sensor data that will be the input to our object detection model. (right) State labels indicating the number of quantum dots at each voltage value: DD = Double Dot, SD = Single Dot, QPC = Quantum Point Contact, SC = Short Circuit.

The idea is to train a model that can take charge sensor data as an input and predict its state labels. To do this, we will leverage Detectron2 and the QFlow dataset.

4.3.2 Training Dataset

It is standard to construct three datasets: training, validation and testing. The training dataset has the data and its true labels for the model to learn on. The validation dataset has the same format but is used after the model is trained to verify its performance. With the model verified, it can be run on test data which has no labels associated with it. Typically, 80% of the data is set aside for training, 10% for validation and the remaining 10% for testing. A custom pipeline was designed to transform QFlow's dataset into a Detectron2 training dataset while allowing the user to customize certain aspects like augmentations, dataset size, training/validation/test data ratios and more. Figure 4.2 shows an example of how an element from the Detectron2 training dataset is constructed from QFlow data.

Ideally, a model would be trained on purely experimental data but only a small percentage of the QFlow dataset is from a real device. In order to improve the ratio of simulated to experimental data, I added the ability for the user to augment their training dataset with various transformations. With these augmentations, simulated data could begin to resemble experimental data. Figure 4.3 shows an example of a random element from the training dataset and one of its augmented counterparts.

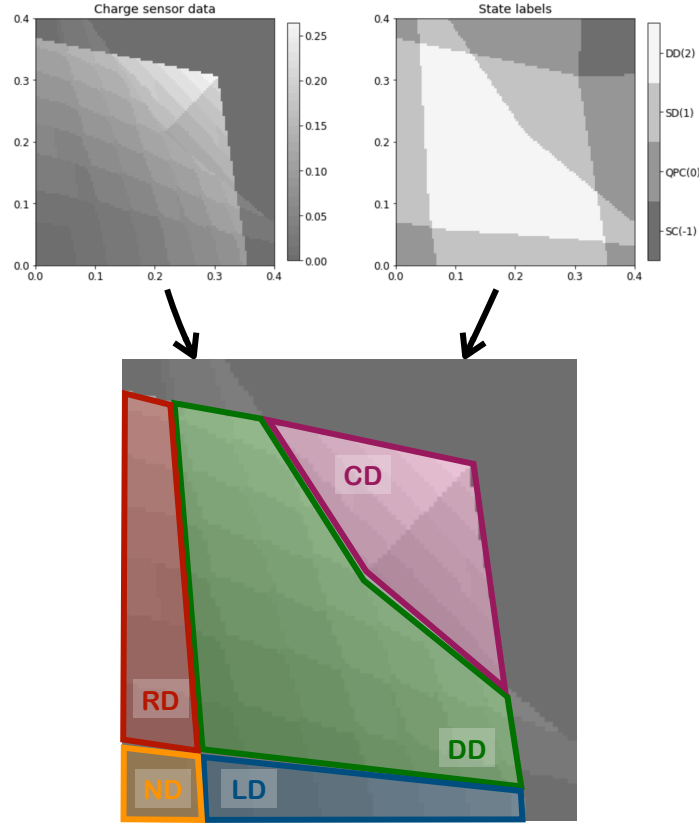


Figure 4.2: Example of how QFlow charge sensor data and state labels (top) can be sent through the dataset creation pipeline to create one element of training data (bottom). The QFlow data is processed to five different class labels each with their own segmentation mask: CD = Central Dot, DD = Double Dot, L(R)D = Left(Right) Dot and ND = No Dot. Experimental data from QFlow had to be manually segmented since no state labels were provided with the data.

Raw



Augmented

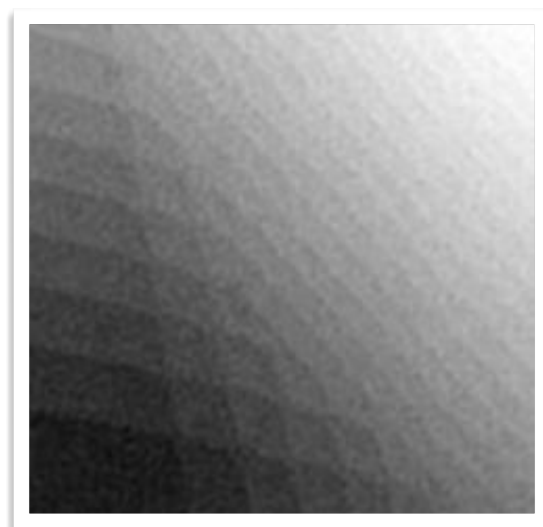


Figure 4.3: Example charge stability diagram from the training dataset that has been augmented with a counter clockwise rotation, cropping, and artificial noise. This aids in generating “artificial” data that closely resembles experimental data one would see from a device.

4.3.3 Results

Two models were trained, each of which tackled a different pillar of automated tuning. The first model segments charge sensor data into voltage regions that exhibit specific dot configurations. The second model also segments charge sensor data but into different voltage regions that exhibit unique charge state configurations.

Coarse Tuning

The goal of coarse tuning is to obtain the voltage ranges that realize a given device configuration (e.g. number of quantum dots). This is achieved experimentally by heuristically identifying features that signal the desired dot configuration. Figure 4.4 shows the performance of the trained model on an image from the validation dataset. The segmentation masks and labels correspond very well with the reference data; verifying the model. Figure

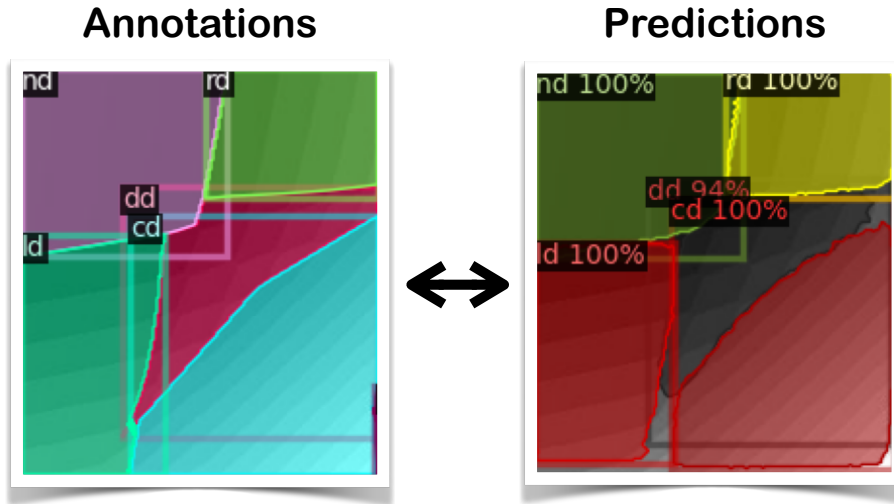


Figure 4.4: Trained coarse tuning model predicting dot configuration regions on an image from the validation dataset. On the left is the image with its true annotations and on the right is the same image passed through the model demonstrating a high degree of accuracy between predicted and actual segmentation masks and labels.

4.5 shows the results of the model applied on the three available experimental images from the test dataset.

Predictions

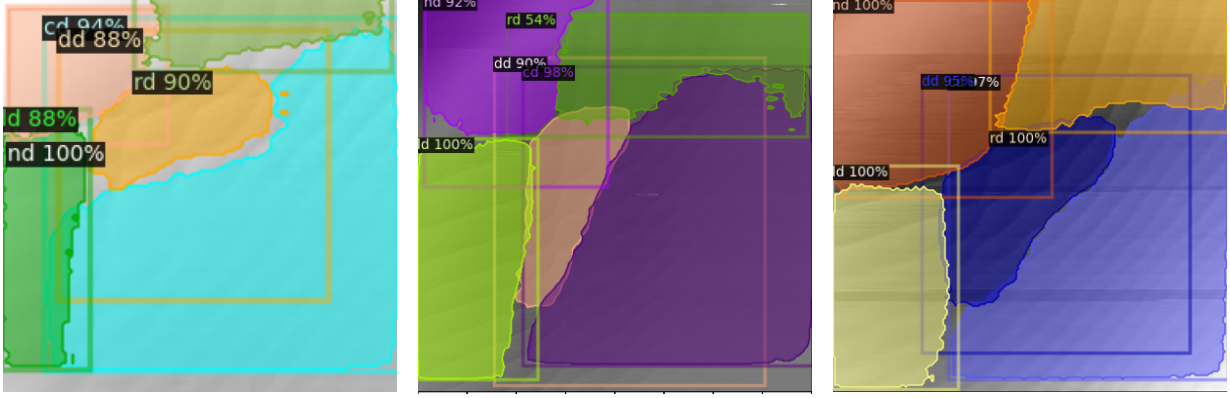


Figure 4.5: Trained coarse tuning model applied to three experimental images from the test dataset. In this case, no annotations were available, but one can see that the model predicts the dot configurations and their locations with a heuristically high degree of accuracy.

It is important to highlight the fact that the model is not only able to segment the voltage regions of interest but also maintain spatial context on where the dots are being formed (left, right, central, etc). The power of this is that the results of inferencing the model can be used downstream in a measurement process to automate the tuning of a device into a desired dot configuration.

Charge Tuning

As a reminder, the goal of charge tuning is to obtain a voltage path that correctly realizes a given charge configuration that from an empty set of dots. For our specific quantum computing platform, the maximum relevant charge configuration is the (2,2) state - two charges in each dot. For this reason, the model was designed to only identify up to this charge state. Figure 4.6 shows the results of the model from an image in the validation dataset. With the model's performance validated, Figure 4.7 shows an image from the test dataset where no annotations are available.

The model is able to correctly inference the data and classify all of the different charge states for simulated data. However, the model struggles a bit for experimental data and perhaps would need to be further trained to see if the performance metrics can be improved.

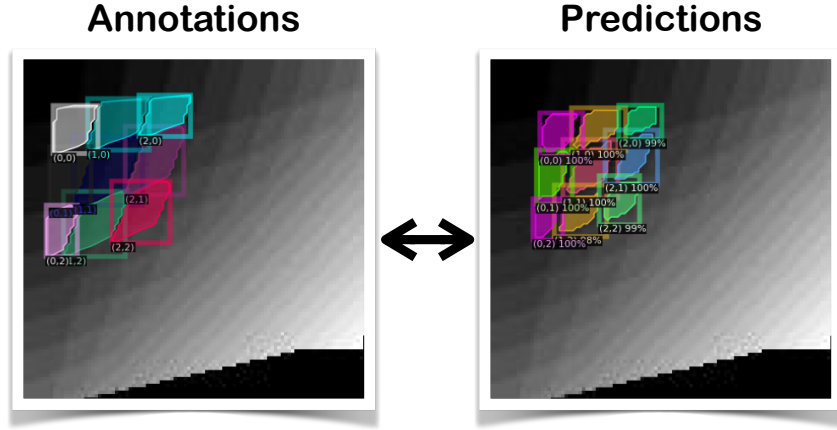


Figure 4.6: Trained charge tuning model applied to an image from the validation dataset. On the left is the image with its true annotations taken from QFlow. On the right is the same image inferred through the model demonstrating a perfect match for expected labels and a high degree of accuracy for predicted segmentation masks. It might be hard to see but the labels are all correct compared to the true labels on the left.

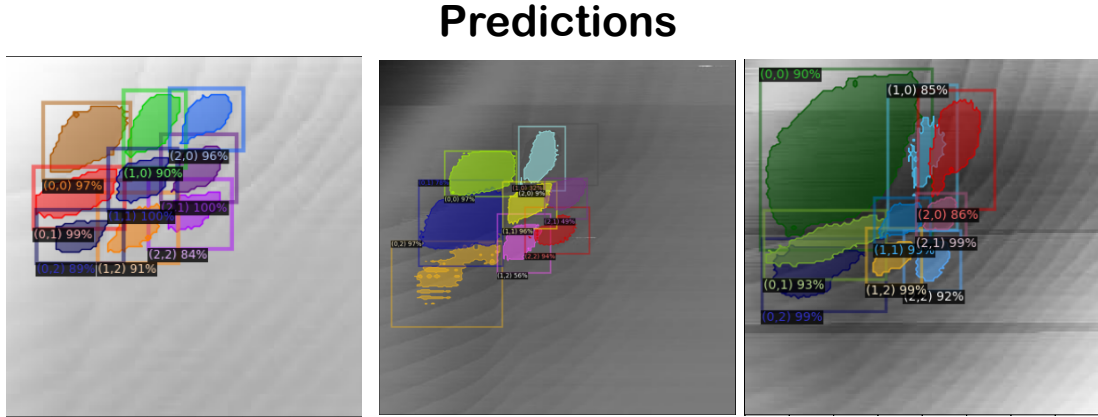


Figure 4.7: Trained charge tuning model applied to the three available experimental images from the test dataset. In this case, no true annotations are available, but the labels for the masks are correct but the segmentation masks aren't perfect.

Assuming the charge states are predicted with a high degree of accuracy, a charge tuning path can be obtained by linking the centroids of the predicted charge state polygons

together to connect the initial and final charge state. This can be seen in Figure 4.8. Although this gives a path in pixel space, it can easily be converted to voltages.

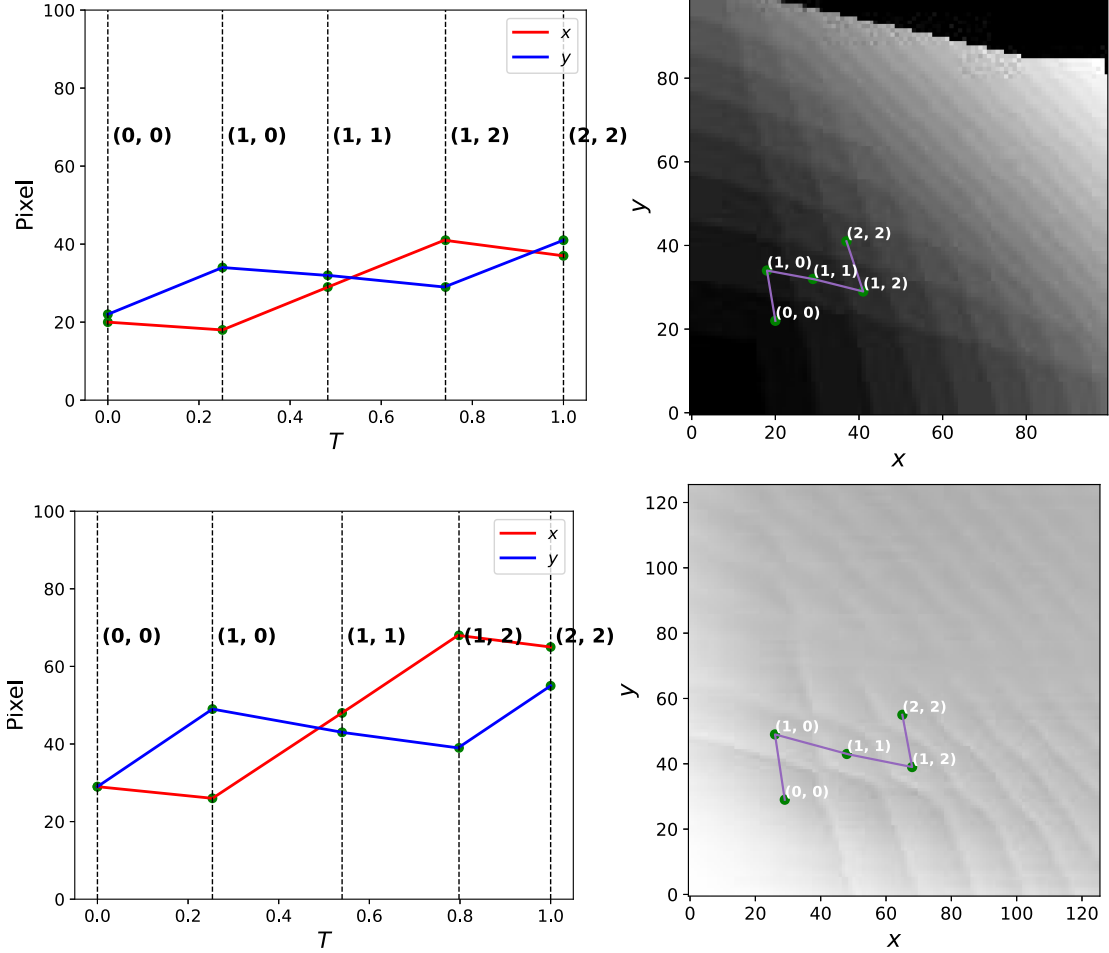


Figure 4.8: Example of a charge tuning path, parameterized by pulse length T , determined from the output prediction of a simulated image (top) and experimental image (bottom). Using the polygon centroids of the predicted charge state regions, and the desired charge path from the user, a path through pixel space can be calculated. Users can then convert their pixel path to a voltage path using the instrument step size used to measure the charge sensor data.

Virtual Gating

The geometry of the predicted centroids provides some information about the capacitive coupling between the quantum dots. Using the centroids of the charge states obtained through charge tuning, we can actually perform a crude virtual gating protocol as well. Figure 4.9 shows the charge paths shown in Figure 4.8 but virtually gated using the relative slopes between three centroids.

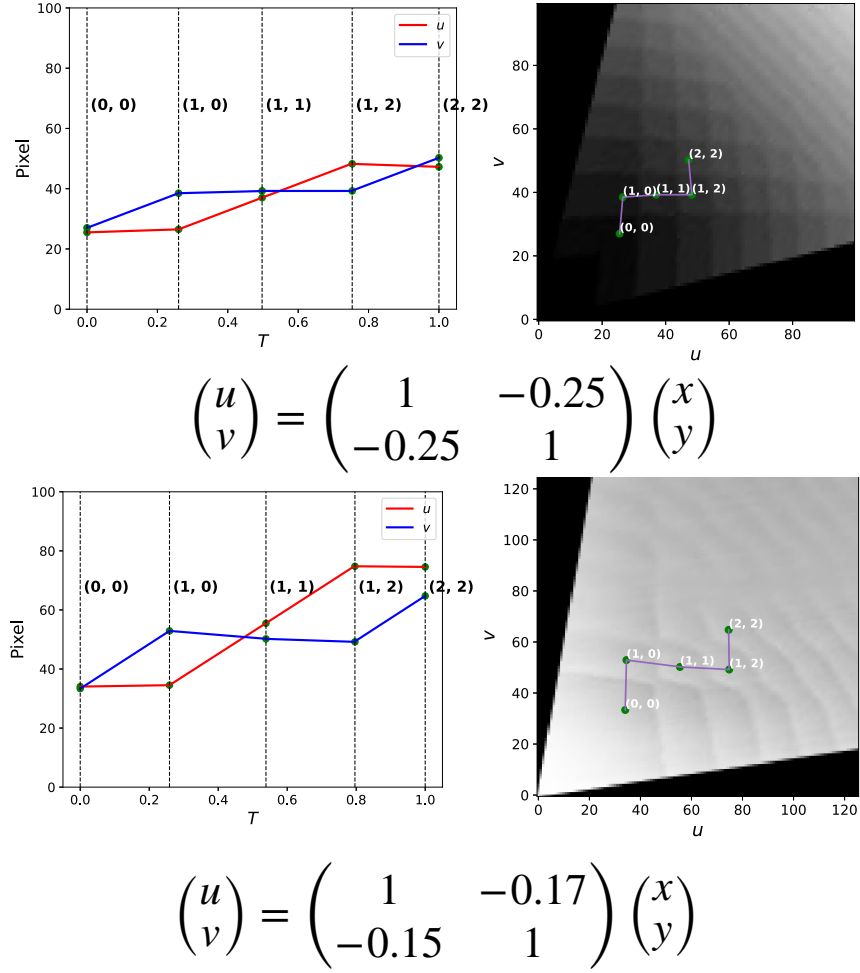


Figure 4.9: Example of a charge path from a simulated image (top) and experimental image (bottom) that has been virtually gated. This is done by using the slopes between neighbouring charge states to construct the virtual gating matrix seen in the image.

One can see that the relative slopes between neighbouring charge states have been nearly eliminated, obtaining a more orthogonal control through the virtual gate space. On the experimental image, even after virtual gating, there are still sources of local slopes. This implies that the virtual gating procedure should be done after every charge transition to account for local changes in the capacitive coupling of the quantum dots.

4.3.4 Performance

While the results are encouraging, further investigation is needed to assess how these models perform relative to established methods in the literature. A crucial benchmark would be evaluating the time it takes for the tuning process; from charge stability diagram measurements to the desired dot or charge configuration. This segmentation architecture, significantly more complex than the lightweight CNN architectures commonly used, may present scalability challenges when integrated into protocols that involve multiple sequential measurements. However, its strength lies in producing spatially segmented regions that are intuitive for human interpretation.

4.3.5 Enhancements

Below is a list of suggested enhancements to expand on this work,

- The models presented should be trained on larger datasets, with data coming from various types of devices and measurement set-ups.
- These models should be embedded into experiment by using the model predictions to aid in a tuning loop. This was not possible as we did not have any double quantum dot devices to tune up.
- A model should be trained to predict dot and charge configurations from direct current measurements. Sometimes charge stability diagrams are obtained through direct current measurement as opposed from charge sensor data.

4.3.6 Summary

In this section, the two models that address coarse and charge tuning of quantum dot devices were presented. The models performed well on the validation and test datasets built from the QFlow dataset.

These contributions suggest that the Mask-RCNN architecture can be a strong candidate in the field of automated tuning. Furthermore, trained models like these can and should be integrated into an data measurement and analysis pipeline in order to assist the tuning of quantum dot devices.

4.4 Automated Tuning of a Single Quantum Dot

4.4.1 Device Under Test (DUT)

Figures 4.10 and 4.11 show the schematic layout of the accumulation mode, n-type device under test. It is important to note that software is able to handle either accumulation or depletion mode devices as well as either p- or n-type doping in the substrate (or any such combination).

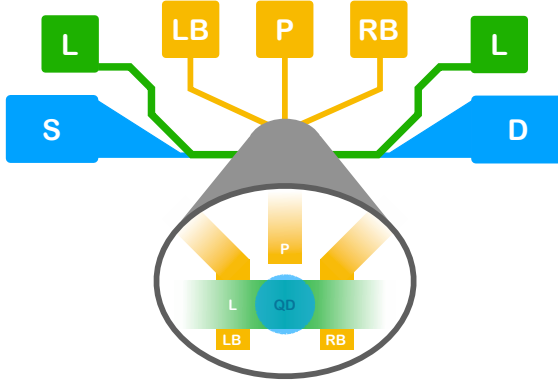


Figure 4.10: Top-view of the lateral gate defined geometry on the surface of the heterostructure. This schematic demonstrates the fan-out of gates from active region to macroscopic wire bonding pads. S and D are the source and drain Ohmic contacts respectively (responsible for setting the Fermi level and reading out the current), L is the lead gate (primarily responsible for 2DEG formation), LB/RB are the left and right barrier gates (used to deplete the 2DEG to confine a quantum dot) and P is the plunger gate (used to tune the electrochemical potential of the dot). The lateral area of the desired quantum dot region is designed to be $80 \times 80\text{nm}^2$.

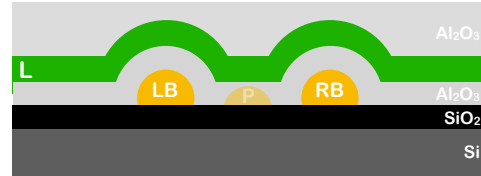


Figure 4.11: Side-view cross section of the device showing the heterostructure layering of materials as well as the relative position of the channel gates. The quantum dot would be ideally formed between the two barrier gates at the interface of the Si and SiO₂ layer. The thickness of each layer is on the order of a few nanometres.

4.4.2 Measurement Set-Up

All the measurements in this section were done through a custom measurement protocol that leverages QCoDeS as its backend data collection software. QCoDeS is the industry standard Python-based data acquisition framework for quantum computing [40]. The software is designed to work with any laboratory equipment and so the set-up in Figure 4.12 is intentionally high-level. For this experiment, the voltage source was a Stanford Instrument SIM900, with an Agilent 34401A as the digital multimeter for current readout. The DUT was cyrogenically cooled to 1.6 K during measurements.

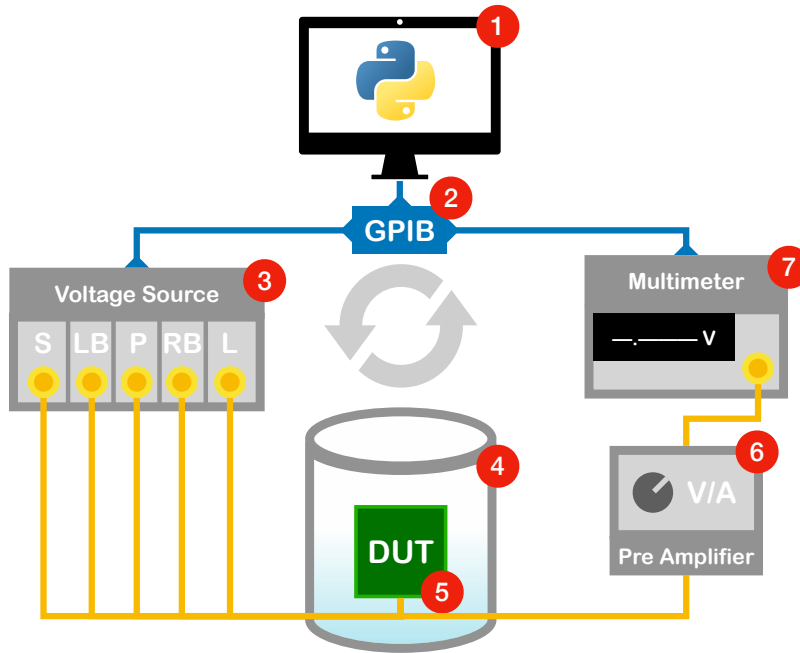


Figure 4.12: High-level overview of the measurement set-up required to execute the tuning protocol. (1) The software is run on the laboratory computer and communicates via (2) GPIB to all of the instruments. (3) The voltage source is connected through (4) the cryogenic fridge to (5) the device. The current is read out from the drain ohmic being connected to a (6) current-to-voltage pre-amplifier which is then connected to a (7) digital multi meter for readout.

4.4.3 High-Level Overview

The software is in its early stages; currently the user has the ability to interrupt measurements stages in case manual tuning is required. The goal of the software is to do most of the work for the user by exploring the complex parameter space. A user can then further refine the device on their own accordance and explore the interesting physics of the device.

The software is built from three important configuration files that the user must design and provide. The tuner can be created with the command,

```
qdtuner = QDTuner(  
    config='path/to/config.yaml',  
    tuner_config='path/to/tuner_config.yaml',  
    station_config='path/to/station_config.yaml'  
)
```

A database file is automatically created, safely storing and backing up measurement data. A log file containing all program output is also created for the user and is constantly updated.

(1) config.yaml

Device characteristics, constraints (min/max voltages and current) and gate connectivity information is defined in this file. It is also where the user defines how the voltage source is connected to the device gates. All of this information is relevant as it completely determines the state of the tuner and how it will proceed to probe and measure the device. Lastly, this file defines any set-up specific parameters such as pre-amplifier sensitivities, ohmic source voltage dividers, etc.

(2) station_config.yaml

This file sets up the instrument information that QCoDeS needs to communicate with the device. This includes instrument parameters such as step size and sweep rates as well as GPIB address information.

(3) tuner_config.yaml

This file contains a set of default tuner configurations such as what functions are used for fitting to certain I-V measurements.

4.4.4 Forming a Single Quantum Dot

Figure 4.13 shows a high-level overview of the I-V measurements required to form a single quantum dot in an n-type accumulation mode device.

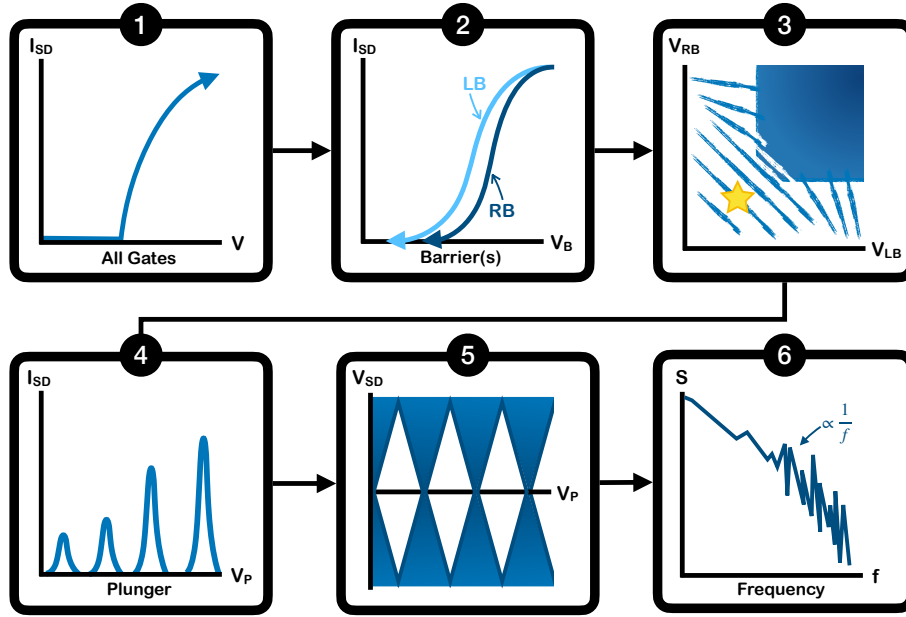


Figure 4.13: The measurements required to tune a single quantum dot for an accumulation mode, n-type device. (1) By biasing the device and sweeping all of the channel gates together towards a positive value, a 2DEG is formed and the device *turns-on*. (2) The barriers defining the dot are tested on whether they can *pinch off* the current in the channel completely; this is done by individually ramping the barrier gates and measuring the current. (3) By performing a two-dimensional voltage sweep spanned by the barriers (around their pinch-off values), a bias point can be selected (gold star). This point realizes a central dot in the device. (4) If the bias point is successful, Coulomb oscillations will be visible upon sweeping the plunger gate voltage at this voltage configuration. (5) By sweeping the plunger and the device bias together, well-defined Coulomb diamonds should form. The diamonds can be processed to estimate dot properties. (6) Noise characterization can be done by looking at the power spectral density curve (optional).

The measurements shown in Figure 4.13 can then be joined together into a protocol that can be run from top to bottom. Figure 4.14 demonstrates what the software is doing when forming a single quantum dot.

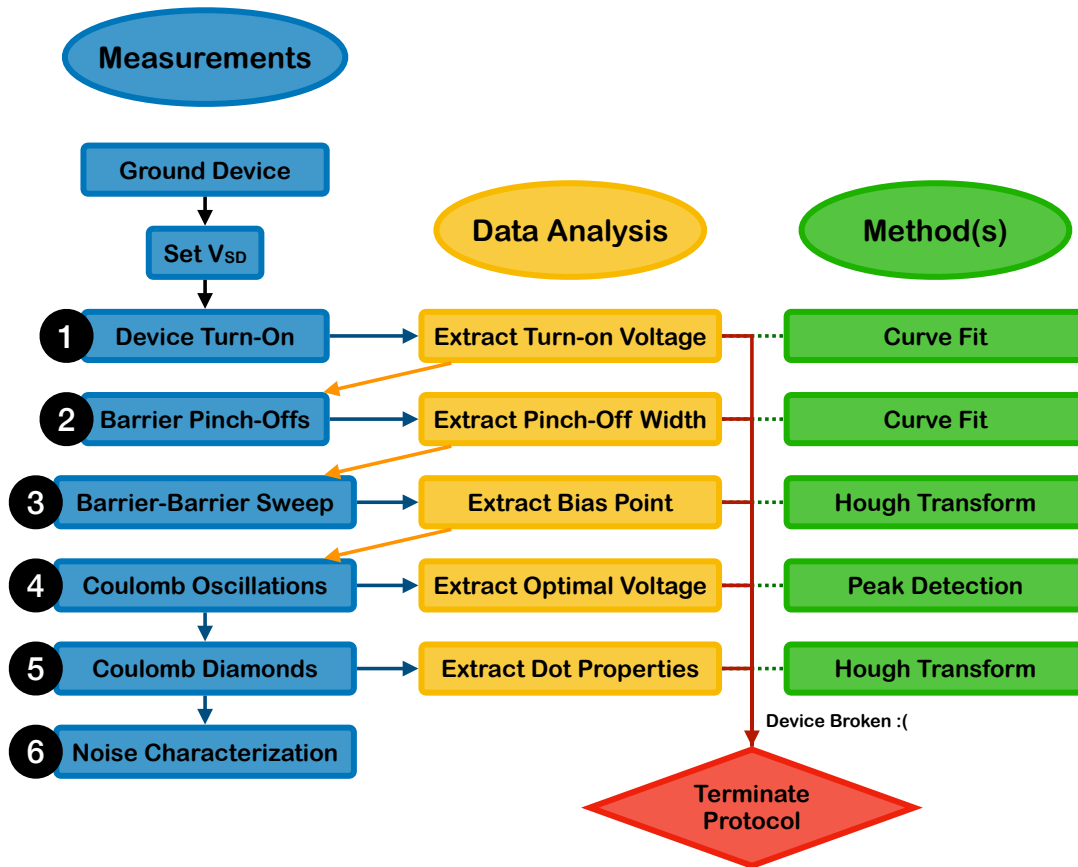


Figure 4.14: High-level overview of the required measurements and data analysis performed in the tuning software. Any results from the data analysis can and will be used as input to the next measurement stage. If any error or anomaly is detected, the tuning software will abort and ground the device; from there a user can take over the tuning process and see what went wrong.

4.4.5 Bootstrapping

To begin, we need to ground the device completely,

```
qdtuner.ground_device()
```

Device Bias

This is followed by applying a small voltage bias to the source ohmic,

```
qdtuner.bias_ohmic(  
    ohmic="S",  
    voltage=0.01 # Volts  
)
```

This is the voltage that is being sent from the voltage source but depending on the divider connected to the ohmic on the device, it will only receive a fraction of it. In this case, the device will receive a standard source drain bias of $100\text{ }\mu\text{V}$ due to the $1/100$ divider connected to the source ohmic.

Global Turn-On

The next step is to check if a current channel can be formed by accumulating a 2DEG in the device; otherwise known as *turn-on*. This is done by sweeping all of the gates along the channel until either: the maximum allowed gate voltage or the maximum current threshold is met,

```
data_T0 = qdtuner.turn_on(  
    minV = 0, # Volts  
    maxV = None, # Volts, if None sweep until max voltage (3 V) or max  
                current (1.5 nA) is measured  
)
```

Having set the maximum current threshold to $I_{\text{max}} = 1.5\text{ nA}$, the program successfully terminated the measurement. The data is then fit to a logarithmic curve to extract properties such as the turn-on voltage and the saturation voltage as seen in Figure 4.15 [35]. This is stored as metadata in the software and will be used for future measurements. The calculated device resistance is $R = 62\text{ k}\Omega$ based on the device bias and the current measured.

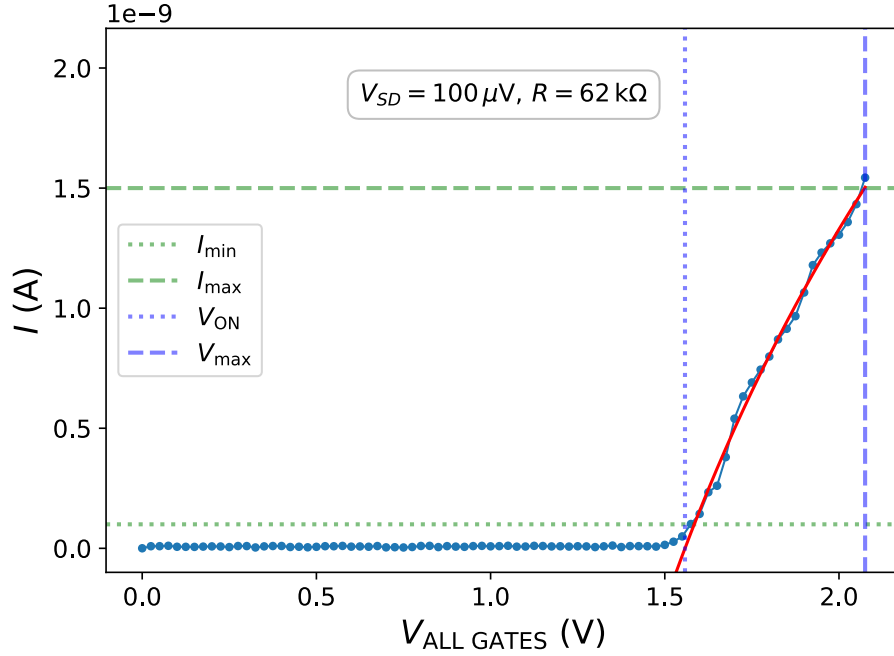


Figure 4.15: Global turn-on curve of the accumulation mode, n-type DUT. The voltage on all of the channel gates are increased to accumulate a 2DEG in the channel. The measurement was automatically terminated once the maximum current threshold of $I_{\max} = 1.5$ nA was measured. The data was then fit to a logarithmic curve to extract the turn-on properties, including turn-on and saturation voltage.

Pinch-Offs

With the device able to turn-on, the next step is to determine whether the barrier gates along the channel are able to pinch off the current. This is a crucial step as these finger gates are the most susceptible to ESR damage or poor alignment. If at this stage a gate is unable to pinch-off, it may be rejected from further tuning.

```
data_P0 = qdtuner.pinch_off(
    gates = ["LB", "RB"]
)
```

The program independently ramps the barrier gate voltage towards zero until either: the current in the channel drops below a minimum threshold or the maximum adjacent gate differential is reached [35]. Once either of those conditions are hit, the voltage is then

slowly brought back to its original voltage. The results of this stage are shown in Figure 4.16.

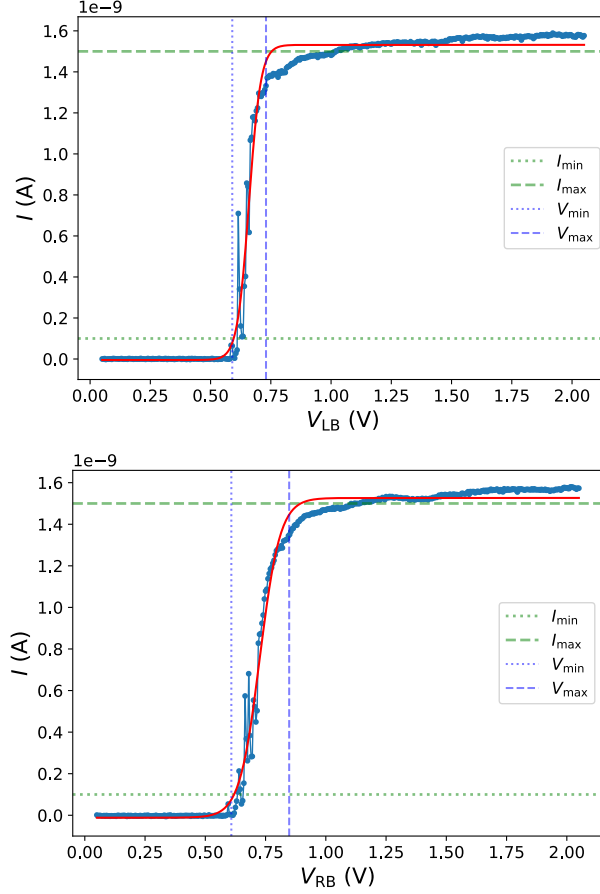


Figure 4.16: Pinch-off curves demonstrating each barriers ability to pinch-off the 2DEG that has been formed. The measurement is done by independently ramping each barrier gate voltage from towards zero. Each curve is fit to a sigmoid and the fit parameters are stored as metadata to be used later in the tuning process. The oscillations seen are due to quantum tunneling events when the device is in between the on and off state.

At this point, the bootstrapping stage would be considered successful; the device is able to turn-on and the barrier gates are able to individually pinch-off the 2DEG that is formed.

4.4.6 Coarse Tuning

With a successful bootstrapping stage, we can begin to form a central quantum dot between the left and right barrier gates.

Barrier-Barrier Sweep

To determine what barrier gate voltages form an central quantum dot, a two-dimensional I-V measurement is performed [35]. To save significant time (and automate the measurement), the sigmoid width of the pinch-off curves is used for the domain of the sweep (since we know this is where confinement and tunneling will occur),

```
data_BB = qdtuner.sweep_barriers(  
    B1 = "LB",  
    B2 = "RB",  
)
```

Figure 4.17 shows the data with clear Coulomb oscillations near barrier pinch-offs indicating the confinement of a lateral quantum dot somewhere between the barriers. The slant of these lines give us an understanding of the lateral location of said dot (as the slant depends on the ratio of the capacitive coupling of the dot to the source and drain). The oscillations occur at nearly 45 degrees to the horizontal; indicating that the dot has been formed exactly between the two barrier gates as desired.

We can apply the Hough transform to identify Coulomb oscillations of interest to determine a set of valid bias points (see Appendix A.1.1 for more details),

```
(LB_bias, RB_bias) = qdtuner.extract_bias_point(  
    data_BB  
)
```

In this case, a bias point was selected at random to use and is shown in Figure 4.17 as the black star. With the barrier gates fixed at this voltage, a central quantum dot should be formed in the device.

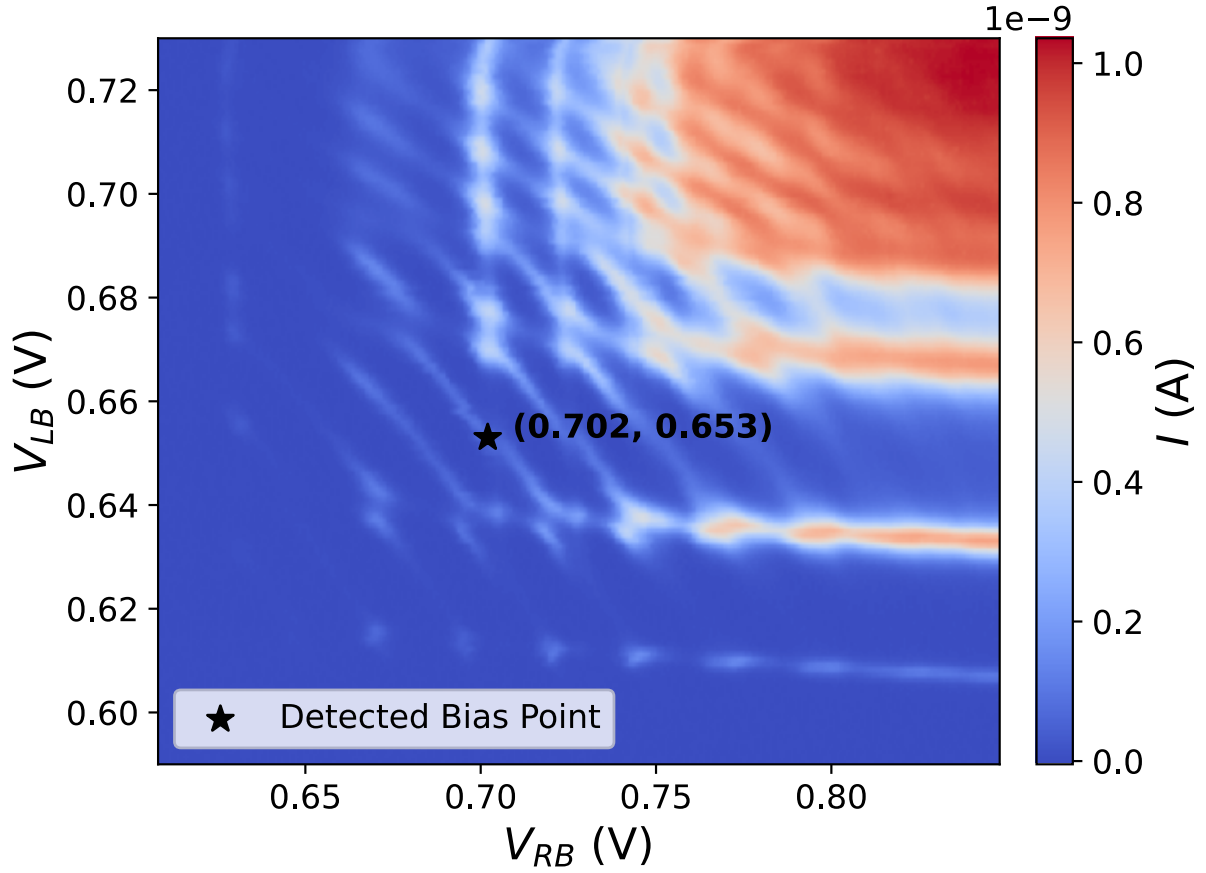


Figure 4.17: Two-dimensional I-V measurement used to determine the barrier voltages that realize a single central quantum dot. The Hough transform was used to detect Coulomb oscillations and a bias point along those lines was selected (black star). The domain of this scan came from the pinch-off widths of the barriers; as that is where quantum behaviour will occur.

Coulomb Oscillations

To benchmark whether or not the bias point was successful; we can now sweep the plunger gate in order to look for Coulomb oscillations,

```
data_CO = qdtuner.coulomb_oscillations(  
    gate = "P",  
    gate_bounds = (-0.5, 0.5), # Volts  
    voltage_configuration = {  
        "LB": LB_bias, "RB": RB_bias  
    }  
)
```

From the data obtained in Figure 4.18, we can see that strong coulomb oscillations are present indicating a single central dot has been formed!

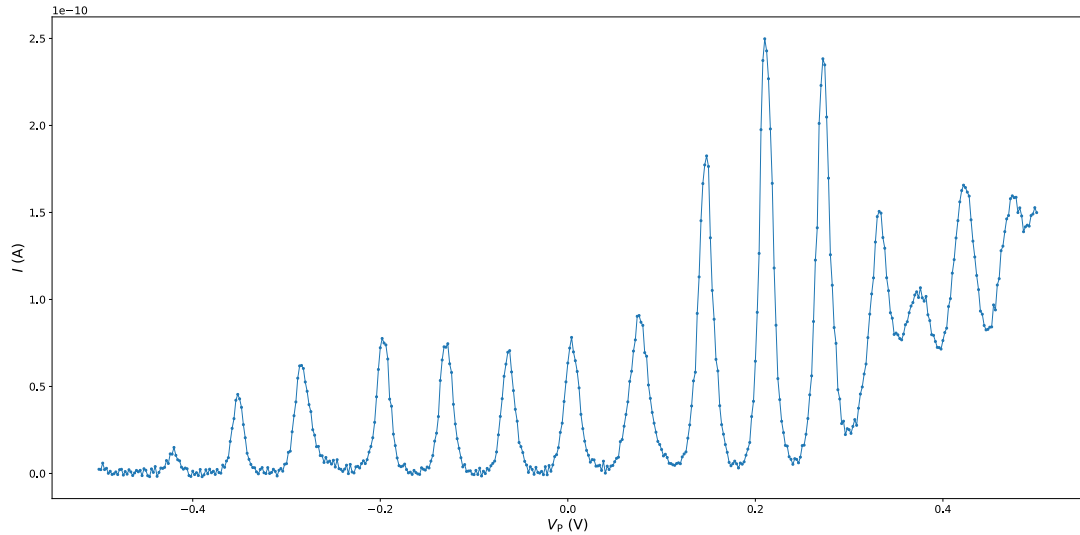


Figure 4.18: Coulomb oscillations obtained by sweeping the plunger gate voltage with barriers fixed at a detected bias point. In this case we can see well defined, evenly spaced Coulomb peaks indicative of a quantum dot having been formed.

Optionally, we can analyze the Coulomb oscillations to extract flank points and sort them based on their sensitivity. This is crucial if we want to set up the quantum dot to act as a charge sensor,

```

result = qdtuner.extract_max_conductance_point(
    data = data_C0
)
P_high_sens, G_high_sens = result["high"]

```

which returns the data shown in Figure 4.19.

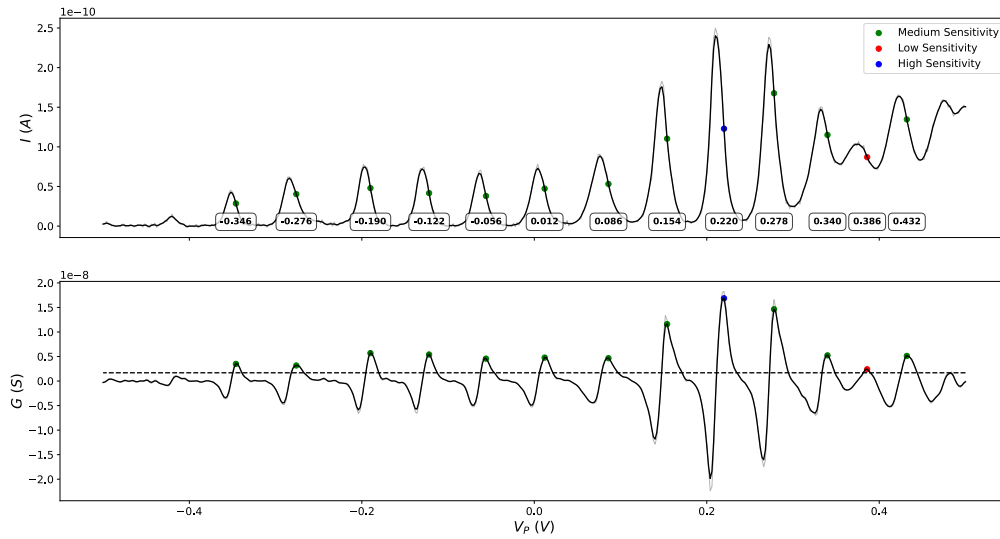


Figure 4.19: Processed Coulomb oscillations to extract flank points and sort them based on sensitivity (low, medium or high).

The code calculates the conductance by taking transconductance along the plunger gate voltage to detect conductance peaks. These peaks can then be sorted in terms of their magnitude and processed into three categories: high, medium and low sensitivity. This information can further guide the tuning process; if the user wants to turn the dot into a charge sensor, the plunger gate should be set to a flank voltage with high sensitivity.

Coulomb Diamonds

With a quantum dot formed, we can conduct Coulomb diamond measurements to get an idea of the dot's characteristics. To do this we invoke the command,

```
data_CD = qdtuner.coulomb_diamonds(  
    ohmic = "S",  
    gate = "P",  
    ohmic_bounds = (-0.5, 0.5), # Volts  
    gate_bounds = (-0.4, 0.1), # Volts  
    voltage_configuration = {  
        "LB": LB_bias, "RB": RB_bias  
    }  
)
```

which returns the data shown in Figure 4.20. Conventionally, this data is processed heuristically to approximate certain quantum dot characteristics such as device capacitance values, estimated dot size, etc.

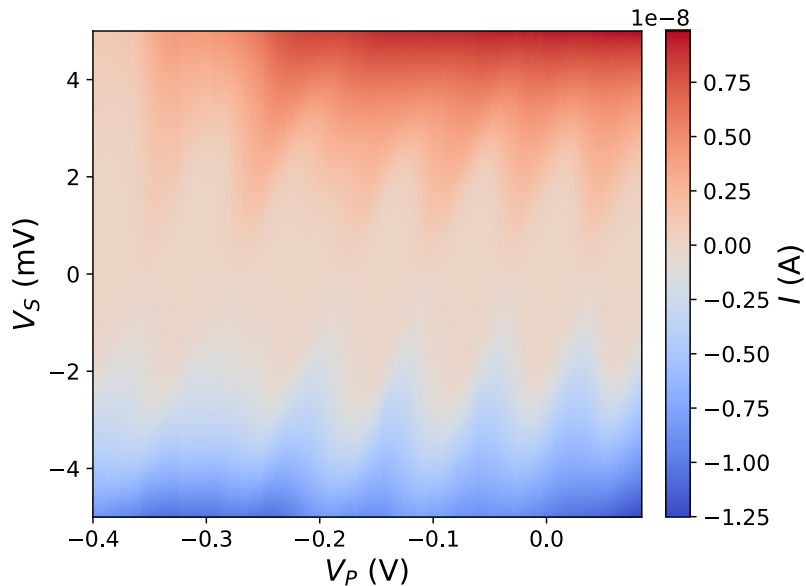


Figure 4.20: Measurement data showing the formation of Coulomb diamonds. As the plunger gate becomes more negative the channel becomes pinched off resulting in a diamond that won't close. Although, as the plunger gate becomes more negative, uniform, well-defined diamonds are formed.

With my `diamondminer` software, the Coulomb diamonds present in the data can be extracted,

```
miner = diamondminer.Miner(
    gate_data=P_data,
    ohmic_data=S_data,
    current_data=I_data
)
diamonds_info = miner.extract_diamonds()
```

which once again performs a Hough transform to detect a set of diamonds (see Appendix A.2 for more details). We can also estimate the electron temperature in the device, by fitting the Coulomb oscillations at the low-bias regime,

```
ohmic_value = 100e-6 # Volts

temperatures = miner.estimate_temperatures(
    diamonds,
    ohmic_value,
    temperature_guess = 2 # K
)
```

The visual results are shown in Figure 4.21 with the estimated device properties provided in Table 4.1.

Parameter	Symbol	Value	Unit
Lever Arm	α	33.6(0.4)	meV/V
Addition Voltage	V_{add}	74.7(0.3)	mV
Charging Energy	V_C	2.50(0.02)	
Total Capacitance	C_Σ	63.9(0.4)	aF
Gate Capacitance	C_G	2.14(0.01)	
Source Capacitance	C_S	17.7(0.3)	
Drain Capacitance	C_D	43.4(0.2)	
Dot Radius	r	76.7(0.3)	nm
e ⁻ Temperature	T_e	2.0(0.1)	K

Table 4.1: Device properties extracted from the Coulomb diamonds measurement in Figure 4.20. The quantum dot area on the device should have a radius of $r \approx 40\sqrt{2}$ nm, indicating the dot formed is slightly larger than the device’s intended design.

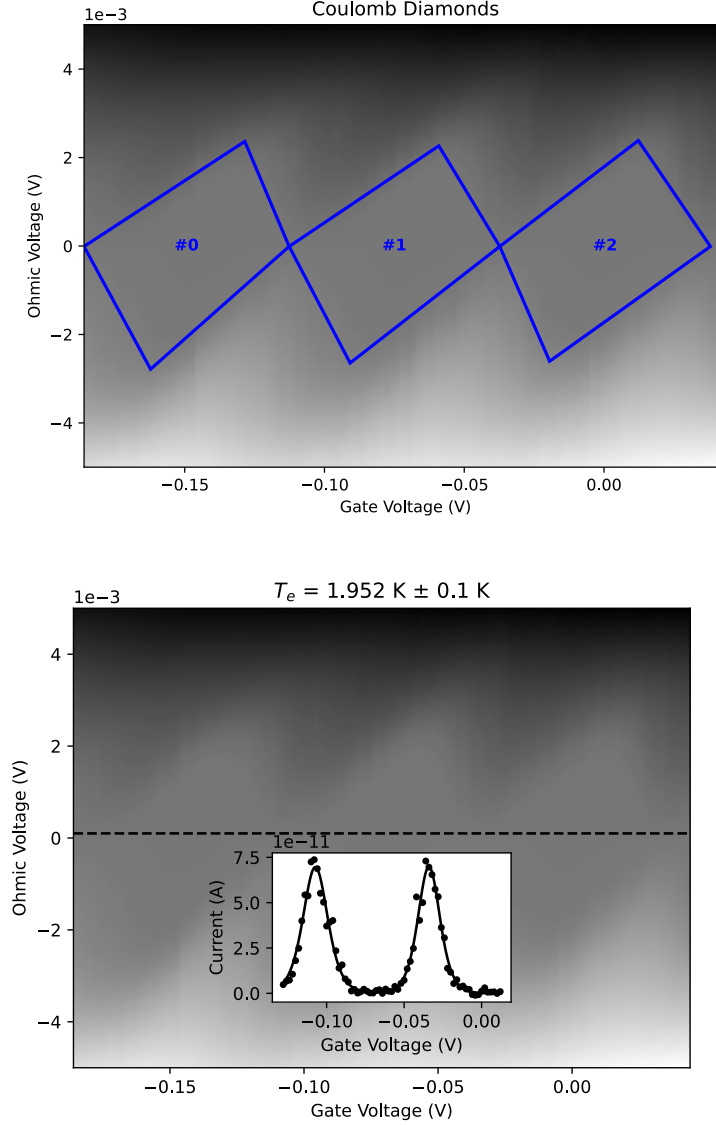


Figure 4.21: (top) Detected Coulomb diamonds (0,1,2) from a cropped region of the measurement in Figure 4.20 that showed the clearest diamonds. From the geometry of each detected diamonds, device properties can be estimated informing the user on the characteristics of the quantum dot formed, as shown in Table 4.1. (bottom) Temperature estimation based on Coulomb oscillations taken at $V_{SD} = 100\mu V$. At this low-bias regime, the temperature of the charges can be estimated from fitting the data to their theoretical behaviour.

4.4.7 Noise Characterization

This stage is completely optional and is not critical to tuning up a quantum dot. To characterize a dot's susceptibility to noise we can fix the plunger gate sensitivity, open up the bandwidth in the set-up, and acquire the data for an extended period of time [58]. This enables the calculation of the current noise power spectrum of the device. As a proof of concept, basic noise characterization was performed on this device. The set-up was limited to a bandwidth of roughly 40 Hz which is rather slow. Once the measurement set-up is prepared, capturing the data can be done with the command,

```
data_I, data_PSD = qdtuner.current_noise_spectrum(  
    t_capture = 1500, # Seconds,  
    voltage_configuration = {  
        "P": P_selected # Volts  
    }  
)
```

This was done for three different plunger gate sensitivities - low, high and Coulomb blockade - resulting in the data shown in Figure 4.22.

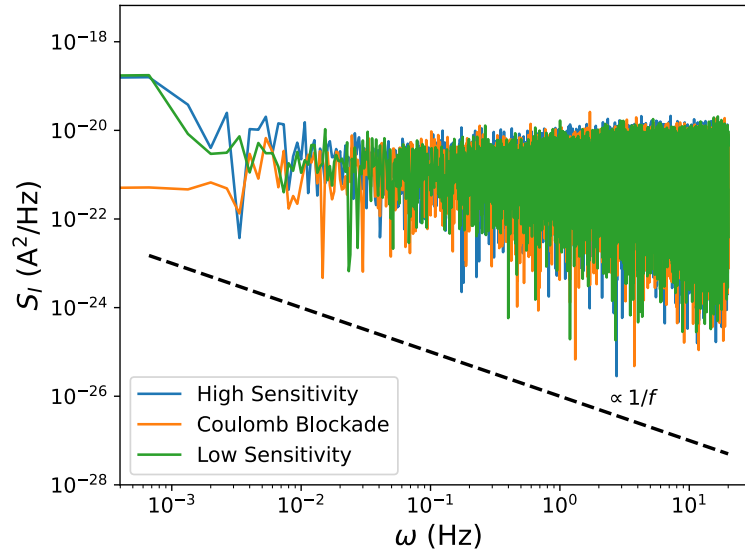


Figure 4.22: Current power spectral density for three different plunger gate sensitivities. As expected, the noise follows a $1/f$ spectrum when not at Coulomb blockade.

Unfortunately, the set-up was rather noisy and so much of the higher frequency data is lost in noise. However, there is an indication that the noise follows the expected $1/f$ relationship. The Coulomb peaks shifted a lot during this measurement likely due high temperature. This meant that the chosen sensitivities were not constant with time, especially over the long measurement duration. In practice, the lower sensitivity plunger gate value should have a spectrum that is somewhere in between the two. Although, as a proof-of-concept this shows that the code works and should be applied to a quieter set-up to get good results.

4.4.8 Enhancements

Below is a list of suggested enhancements to expand on this work,

1. The tuning software needs to be designed as a general quantum dot device tuner. Right now, it can handle single quantum dot geometries but in the future it needs to be extended to multi-dot (2+) systems.
2. A machine learning classifier could be designed to classify whether a bias point is “good”. Right now, a suggested bias point is picked as a candidate at random and the Coulomb oscillations that follow are heuristically determined to be good or not. This would require constructing a dataset that exemplifies good and bad Coulomb oscillations to be used as a predictor for bias point success.
3. A machine learning classifier could be designed to classify device performance based on the bootstrapping (turn-on and pinch-off) curves that are measured. For instance, if a barrier gate is unable to pinch-off, it is defective; thus the device is grounds for rejection. Ideally, a classifier should determine a device quality factor before the protocol continues to coarse tuning; saving valuable time and resources.

4.4.9 Summary

In this section, a completely automated tune up of a single quantum dot was demonstrated. Starting from a grounded device, a single quantum dot was formed through the analysis of various I-V measurements. Furthermore, the Coulomb diamonds indicating proper dot formation were analyze to predict device properties. These properties align closely to the expected properties based on the device geometry.

Chapter 5

Optimal Control of Semiconductor Spin Qubits

5.1 Motivation

To realize algorithms in any quantum computing platform, the Hamiltonian must be engineered in a precise and controlled way through the use of experimental control pulses. Since its inception, the Gradient Ascent Pulse Engineering (GRAPE) method has been the dominant method of designing control pulses that yield high fidelity gate operations [24]. Modern machine learning tool-kits and model deployment on Graphics Processing Units (GPUs) have allowed for machine learning to become a standard tool for designing control pulses. This field remains an active area of research with optimization protocols being developed and deployed on all quantum computing platforms; with some applications beginning to emerge for silicon spin qubits [59, 31, 53]. With our group fabricating more and more devices, the need for a tool to design unitary gates with high fidelity is of great importance.

The work in this chapter highlights the design and implementation of a unitary gate design protocol on physically simulated semiconductor spin qubits. The protocol also accounts for experimental limitations such as sources of spin decoherence, bandwidth and amplitude constraints. My work leverages PyTorch, a modern machine learning tool-kit commonly used in optimization problems [41].

5.2 QuDiPy Simulator

QuDiPy, our group's software, is capable of simulating the behaviour of spin qubits in realistic device geometries [16]. The goal of this software is to be a general purpose tool for simulating electron dynamics in quantum dots with an emphasis on quantum information processing.

As a reminder, for an open quantum system we have the master equation,

$$\frac{d}{dt}\rho_S(t) = -i[\mathcal{H}_S(t), \rho_S(t)] + \mathcal{L}(t). \quad (5.1)$$

To motivate the optimal control problem, we need to define $\mathcal{H}_S(t)$ and $\mathcal{L}(t)$ for the simulated system. QuDiPy assumes a device layout shown in Figure 5.1.

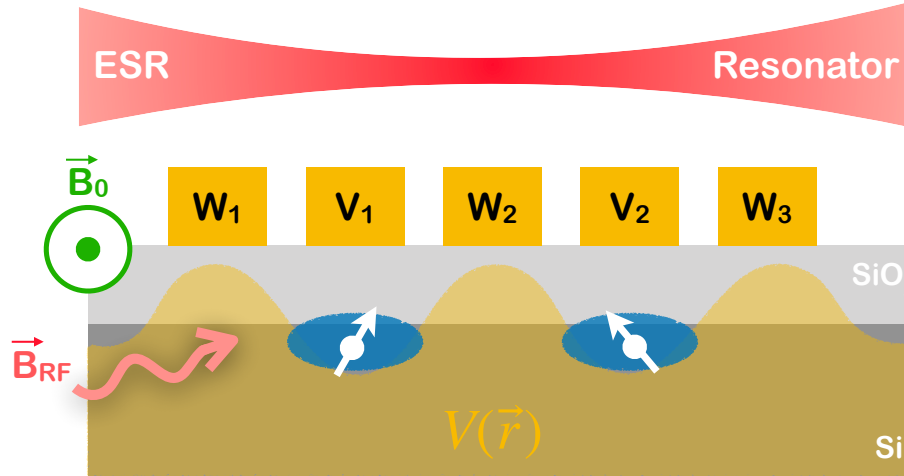


Figure 5.1: Example geometry of a double quantum dot SiMOS device with one spin qubit in each dot. External magnetic fields come from an ESR resonator near the device and the potential inside the heterostructure arises due to the voltages applied onto the surface gates ($\vec{V}$ and $\vec{W}$). Each of the quantum dots are assumed to have no more than one spin qubit each.

5.2.1 Hamiltonian

Let us consider a linear chain of N quantum dots containing no more than one electron each. The system Hamiltonian in the lab frame comes from the energy contributions due

the external static and ESR magnetic field as well as any interactions between neighbouring spin qubits. To simplify the fact that each operator is a Kronecker product of N matrices each with dimension 2×2 we can define,

$$A_k \equiv \left(\bigotimes_{i=1}^{k-1} \mathbb{I} \right) \otimes A_k \otimes \left(\bigotimes_{i=k+1}^N \mathbb{I} \right), \quad \mathbb{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad (5.2)$$

The laboratory frame Hamiltonian is transformed into the rotating wave approximation and is given by,

$$\begin{aligned} \mathcal{H}_S(t) = & \underbrace{\sum_{j=1}^N \frac{1}{2} \left[\left(1 + \frac{\delta g_j(t)}{2} \right) \omega - \omega_{RF} \right] Z_j}_{\text{Zeeman}} + \underbrace{\sum_{j=1}^{N-1} \frac{J_j(t)}{4\hbar} \vec{\sigma}_j \cdot \vec{\sigma}_{j+1}}_{\text{exchange}} \\ & + \underbrace{\frac{\Omega(t)}{2} \left[\cos \phi(t) \sum_{j=1}^N X_j + \sin \phi(t) \sum_{j=1}^N Y_j \right]}_{\text{radio frequency field}}, \end{aligned} \quad (5.3)$$

where $\omega = \frac{2\mu_B B_0}{\hbar}$ is the Larmor frequency and $\Omega(t) = \frac{2\mu_B B_{RF}(t)}{\hbar}$ is the Rabi frequency [25].

If we want to be able to control the spin qubit evolution coherently, we clearly need a way to control the effective parameters: g -factor deviation, δg , of each spin qubit as well as the exchange coupling J between neighbouring qubits. Fortunately, the magnitude of these effective parameters are directly dependent on to the voltages applied to the gates on the device and can be calculated.

5.2.2 Lindbladian

There are two processes contributing to the decoherence of spin states in this system: dephasing T_1 and relaxation T_2 . The Lindblad operators and their associated transition rates for the j^{th} electron are defined as,

$$L_{1,j} = \sigma_{+,j}, \quad \gamma_{1,j} = \sqrt{\frac{p_j}{2T_1}} \quad (5.4)$$

$$L_{2,j} = \sigma_{-,j}, \quad \gamma_{2,j} = \sqrt{\frac{1-p_j}{2T_1}} \quad (5.5)$$

$$L_{3,j} = Z_j, \quad \gamma_{3,j} = \frac{1}{2\sqrt{T_2}}, \quad (5.6)$$

where $p_j = \frac{1}{\exp \frac{\hbar\omega}{k_B T_e} + 1}$ is the statistical probability of finding the j^{th} electron in the spin up state based on the presumed Boltzmann distribution [25]. For that reason, the dissipative part of the master equation is given by,

$$\mathcal{L}(t) = \sum_{\alpha=1}^3 \sum_{j=1}^N \gamma_{\alpha,j}^2 \left(L_{\alpha,j} \rho_S(t) L_{\alpha,j}^\dagger - \frac{1}{2} \left\{ L_{\alpha,j}^\dagger L_{\alpha,j}, \rho_S(t) \right\} \right). \quad (5.7)$$

5.2.3 Device Simulation

Assumed Geometry

In practice, the DUT will have to be designed with CAD and processed by a third-party solver `nextnano++` [4]. This tool, when provided with a device geometry and voltages to simulate over, calculates the three-dimensional potential landscape in the device. With the potential landscape, we can calculate many interesting physical quantities that directly affect the evolution of a spin qubit in a quantum dot.

Potential Landscape Extraction

This potential is then processed by QuDiPy to acquire the two-dimensional horizontal slices of potential energy and electric field above the quantum well interface. QuDiPy can then extract single-electron wavefunctions and the energy spectrum of the system if provided with this information.

g -factor Deviation

The deviation of the g -factor for a given electron depends on the out-of-plane component of the electric field,

$$\delta g_i = \frac{g_i(E_y)}{g_i(E_y = 0)} - 1 = \eta_2 \langle \psi_i | E_y^2 | \psi_i \rangle, \quad (5.8)$$

where η_2 depends on material properties in the heterostructure [25]. The electric field directly comes from the voltages applied on the gates in the system so in principal,

$$\delta g_i = f(\vec{V}, \vec{W}), \quad (5.9)$$

with V_i dominantly contributing to the deviation of δg_i .

Exchange Interaction

The exchange interaction is unique as it requires two-electron spectra to evaluate the interaction energies. This interaction also depends on the surface gate voltages in the system as they are responsible for how the electron wave functions will overlap. At a high-level, the potential is partitioned into pairs where each adjacent pair of dots are masked and the two-electron energy spectrum is calculated. The corresponding exchange interaction for the i th pair is then given by the difference between the ground and first excited state energies [25],

$$J_i = E_{i,i+1,e} - E_{i,i+1,g}. \quad (5.10)$$

The wavefunction overlap directly arises from the voltages from the gates in the system so in principal,

$$J_i = f(\vec{V}, \vec{W}), \quad (5.11)$$

with W_i dominantly contributing to J_i [25]. Although, a large de-tuning between V_i and V_{i+1} can also lead to a significant exchange interaction.

Effective Parameter Interpolation

We wish to optimize over the *experimental* parameters in our set-up (voltages on the gates and the ESR field). However, the Hamiltonian is defined in terms of effective parameters ($\delta\vec{g}$ and $\vec{J}$). If we wish to optimize experimental parameters, we need a computational connection from experimental to effective parameters. To do this efficiently, we rely on QuDiPy's pre-calculated interpolation object that acts as a map from experimental voltages to effective parameters in our system. Figure 5.2 shows an example of how effective parameters are obtained from the voltage signals. The convenience of this object is that if a voltage goes outside the range over which the interpolation object was simulated under, a value for all of the effective parameters can be interpolated.

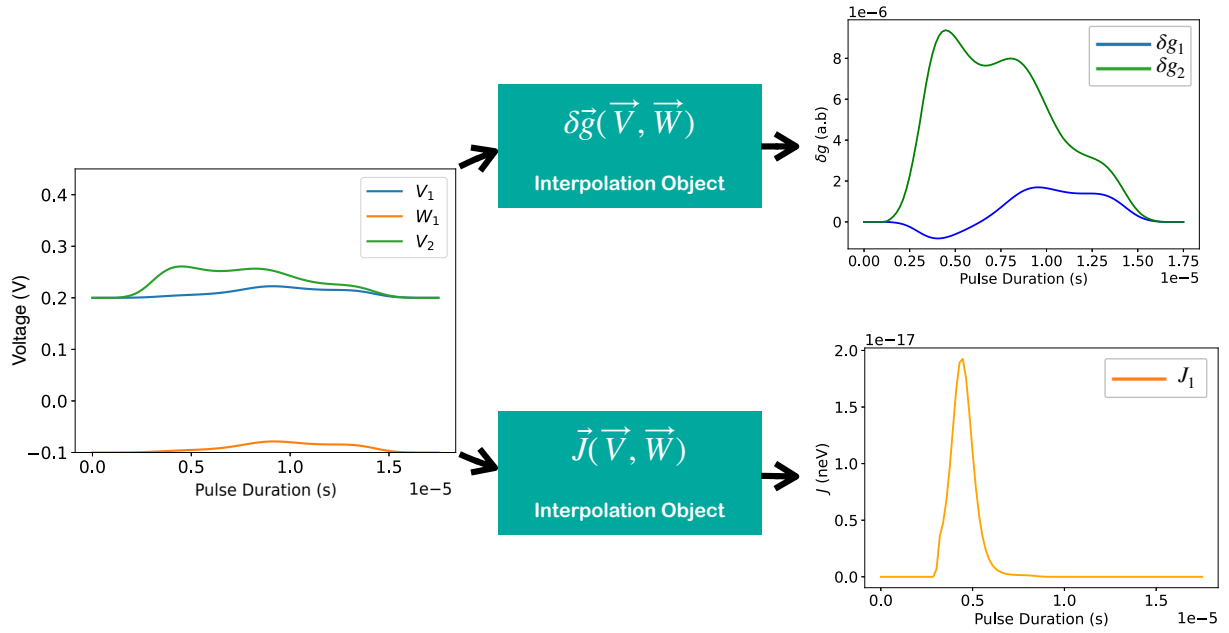


Figure 5.2: Demonstration of how voltage pulses are turned into effective pulses using the QuDiPy interpolation object. In this case, with such a high tunnel barrier voltage between the two qubits the electrons are well isolated and so have essentially zero exchange interaction. Being so well isolated, it explains why the deviation in g -factors so closely resemble the behaviour of the voltage pulses.

5.3 Control Loop Protocol

5.3.1 Overview

Figure 5.3 shows an overview of protocol that was developed.

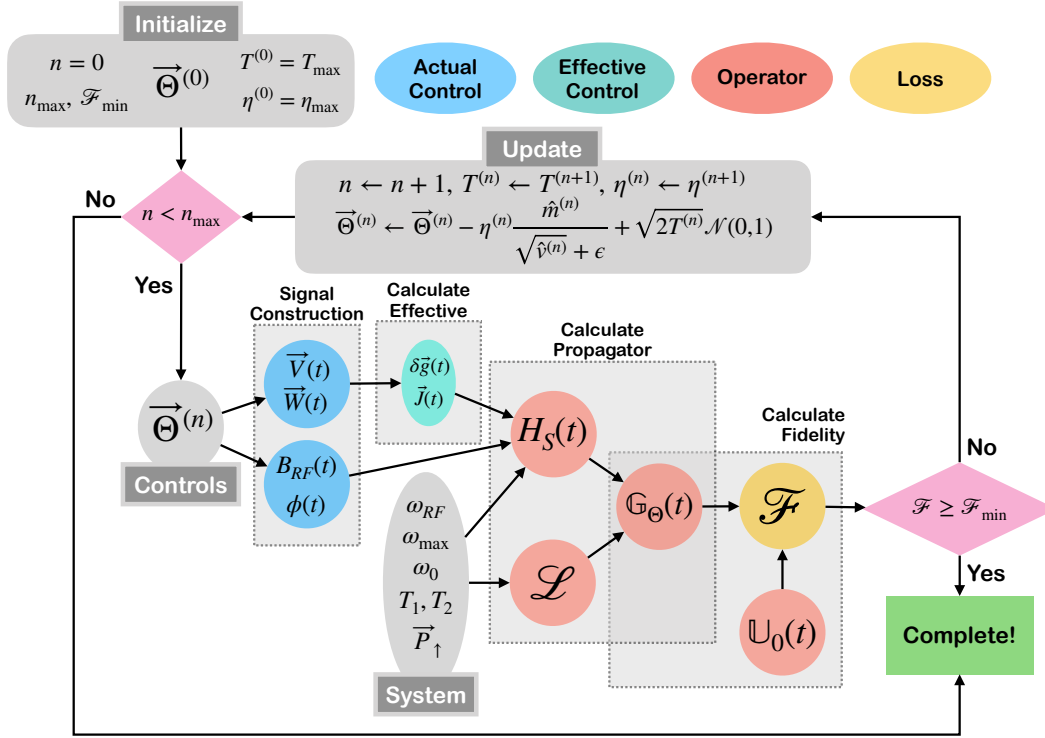


Figure 5.3: Flow diagram of the optimization protocol used for designing experimental control signals that maximize unitary gate fidelity. First, optimization parameters and system values are defined and initialized. The loop will then begin and will consist of signal construction, effective parameter interpolation, propagator construction and fidelity calculation. The parameters are then updated based on a modified Langevin dynamics update rule. This loop will continue until either the minimum fidelity or the number of iterations is met.

5.3.2 Signal Construction

Assuming a linear N dot array, there are $2N + 1$ experimental control signals: (1) plunger gate voltages $V_i(t), i \in [1, \dots, N]$, (2) barrier gate voltages $W_i(t), i \in [1, \dots, N - 1]$, (3) ESR amplitude $B_{RF}(t)$ and phase $\phi(t)$. We wish to parameterize these signals with as few “tunable” parameters as possible in order to manage the computational scaling of this protocol. Figure 5.4 demonstrates the three signal expansions available to the user to realize an arbitrary control signal: Piece-Wise Constant (PWC), Gaussian, or Fourier.

For each of these expansions we assume a time interval $t \in [0, T]$ that we partition the signal into M even time slices $\Delta t = \frac{T}{M}$. The goal is to construct a signal $\mathcal{S}(t) : [0, T] \rightarrow \mathbb{R}$ that represents the pulse we wish to realize experimentally.

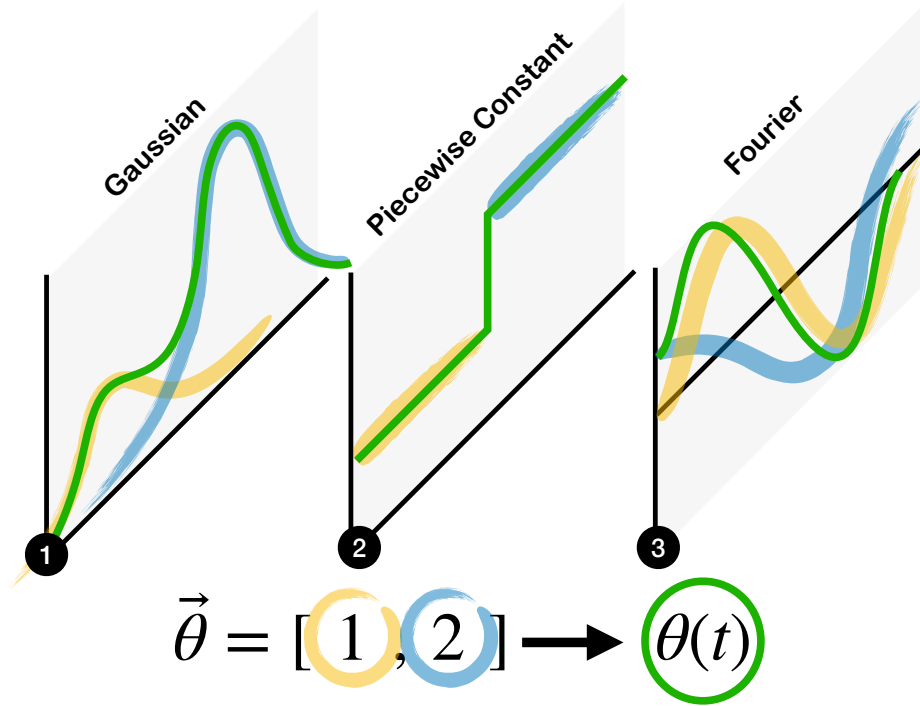


Figure 5.4: The three signal expansions the user can choose from. Each element in the parameter vector $\vec{\theta}$ will map to an individual signal expanded either using the (1) Gaussian, (2) PWC or (3) Fourier expansion. In this example, the value of the element in the vector corresponds to the amplitude of the basis signal that is used in the expansion. The resultant signal (green) is obtained by summing all of the individual expanded signals.

Piece-Wise Constant Expansion

We can parameterize a signal $\mathcal{S}(t) : [0, T] \rightarrow \mathbb{R}$ as a collection of constants $[S_1, S_2, \dots, S_M]$ where,

$$S_m = \mathcal{S}((m-1)\Delta t \leq t < m\Delta t) = \text{const.}, \quad m \in [1, M] \quad (5.12)$$

where M is the number of time steps in the time interval. It is clear that for this kind of parameterization, we have M optimization variables per signal (each of which represent the amplitude for each step): $\vec{S} = [S_1, S_2, \dots, S_M]$. The parameter vector to be optimized would be $\vec{\Theta} = [\vec{S}]$ where $\dim \vec{\Theta} = (1 \times M)$.

Gaussian Expansion

We can build our arbitrary signal by summing lots of individual Gaussian functions. Let's assume we have N Gaussian functions that we wish to turn into a signal. For this, we can tune each of the following parameters: (1) amplitudes: $\vec{\Theta} = [A_1, A_2, \dots, A_N]$, (2) means: $\vec{\mu} = [\mu_1, \mu_2, \dots, \mu_N]$, (3) standard deviations: $\vec{\sigma} = [\sigma_1, \sigma_2, \dots, \sigma_N]$. This gives the most general representation as,

$$S(t) = \sum_{i=1}^N A_i G(t; \mu_i, \sigma_i), \quad G(t; \mu, \sigma) = \exp\left(-\frac{(t-\mu)^2}{2\sigma^2}\right), \quad (5.13)$$

and μ, σ represent the mean and standard deviation of a Gaussian. The parameter vector to be optimized would be $\vec{\Theta} = [\vec{A}, \vec{\mu}, \vec{\sigma}]$ where $\dim \vec{\Theta} = (3 \times N)$. In this implementation, the Gaussians are evenly spaced with a fixed standard deviation and so the only set of parameters to optimize would be the amplitude of each Gaussian.

Fourier Expansion

We can also parameterize an arbitrary signal using the familiar Fourier expansion. We can use N coefficients to represent the signal we wish to optimize,

$$\mathcal{S}(t) = a_0 + \sum_{i=1}^N A_i \cos(\omega_i t) + B_i \sin(\omega_i t), \quad \omega_i = \frac{2\pi i}{N}. \quad (5.14)$$

With this our optimization parameters would be $\vec{\Theta} = [\vec{A}, \vec{B}]$ where $\dim \vec{\Theta} = (2 \times N)$. However, as we will shortly see, we don't really need a_0 because we can control the offset through constraining the minimum and maximum of the signal.

5.3.3 Experimental Limitations

Once a signal has been constructed from a set of optimization parameters, it needs to be constrained to obey experimental limitations. Figure 5.5 visually demonstrates how a signal is constructed and then proceeds to be constrained.

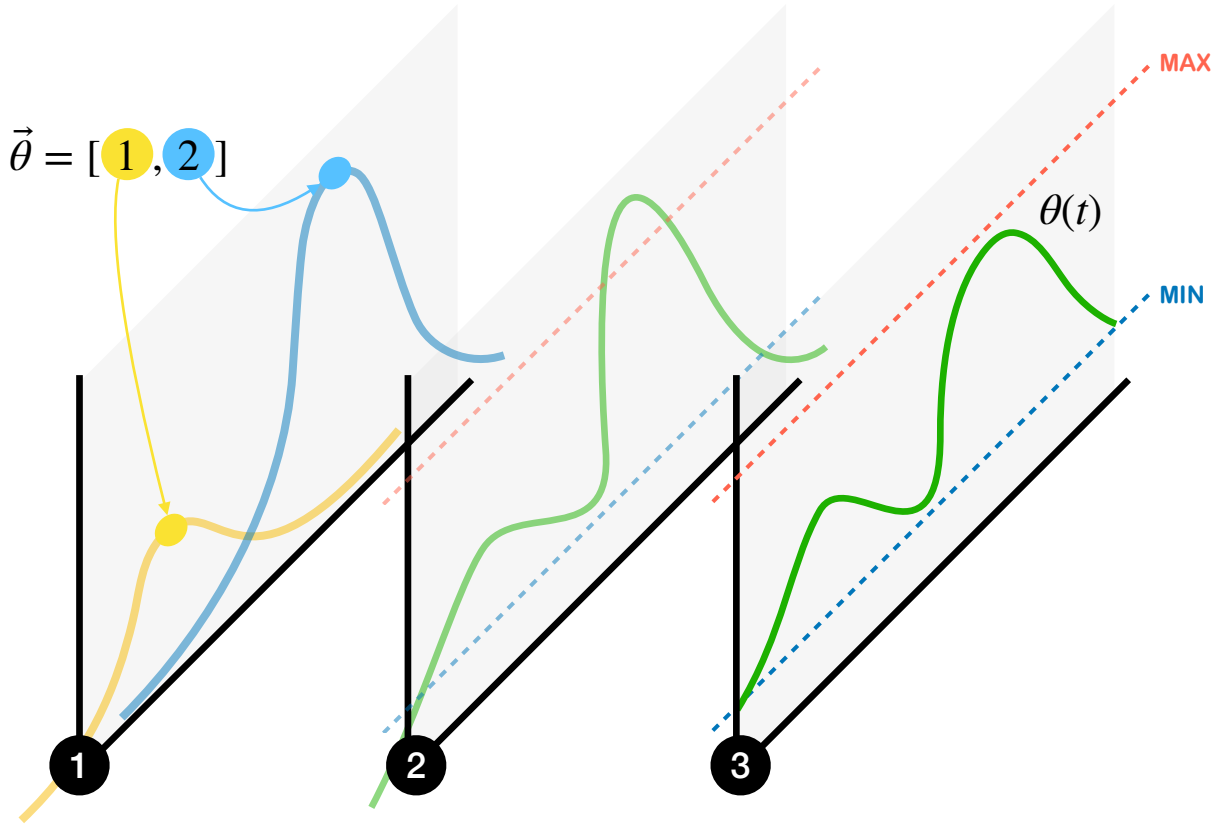


Figure 5.5: Example of a Gaussian signal being expanded and constrained in amplitude and bandwidth. (1) Each element in the parameter vector is converted to individual Gaussian signals (yellow and blue) which are then summed together to create the unconstrained signal (green). (2) Depending on the signal parameters, the resultant signal may exceed pre-defined experimental constraints and (3) is then subsequently constrained to get the final signal $\theta(t)$.

Amplitude

There are a very specific set of amplitude constraints that must be satisfied in these devices,

1. Gate voltages *must* stay within their specified bounds in order to ensure charge stability within the dots (one of the electrons doesn't accidentally hop to/from the neighbouring dot).
2. The power of a field directly depends on its amplitude and so to minimize the RF power delivered to the device, we must constrain the external magnetic field amplitude.
3. The phase of the magnetic field should stay within $\phi \in [-\pi, \pi]$.
4. Any pulse must start and stop at the same value, enabling generated pulses to be strung together to create a quantum circuit.

All of these constraints can be addressed by feeding the signal through a routine before it has been bandwidth limited (once again leveraging automatic differentiation to handle back-propagation) [52],

$$f_{\text{down}}(t, g) = \frac{1}{1 + \exp(gt)} \quad (5.15)$$

$$f_{\text{up}}(t, g) = 1 - f_{\text{down}}(t, g) \quad (5.16)$$

$$W_{\text{amp}}(S_m, l, u, g) = \left[2f_{\text{up}}\left(\frac{S_m - \left(\frac{u+l}{2}\right)}{\frac{u-l}{2}}, g\right) - 1 \right] \left(\frac{u-l}{2}\right) + \left(\frac{u+l}{2}\right), \quad (5.17)$$

where l and u are the lower and upper bound of the pulse's amplitude respectively, g is the window's rate of ascent/descent. We can then define a constraining function, $\mathcal{C}$, that can be modified by user parameters, l, u and g ,

$$\hat{S}(t) = \mathcal{C}(S_m, l, u, g, \Delta t, t) \quad (5.18)$$

$$= f_{\text{up}}\left(\frac{t - \Delta t}{T}, g\right) f_{\text{down}}\left(\frac{t - (T - \Delta t)}{T}, g\right) W_{\text{amp}}(S_m, 0, u - l, g) + l, \quad (5.19)$$

ensuring $\hat{S}(t) \in [l, u]$ for the entire pulse duration. The function parameters, g and Δt , are currently defaulted to satisfactory values but they can be experimented with.

Bandwidth

Laboratory equipment will only have a certain bandwidth of allowed frequencies that are available for arbitrary signal generation. To overcome this hurdle, we can bandwidth limit the pulses *during* the control loop and make use of auto-differentiation to handle any complicated back-propagation of gradients [52]. Assuming the signal is sampled at a rate f_s , the maximal Fourier component number $N_{\max}$ can be derived as,

$$N_{\max} \leq \lfloor N \frac{f_{\max}}{f_s} \rfloor. \quad (5.20)$$

During the optimization phase, we transform the pulse sequence from the time domain into frequency domain using the discrete Fourier transform,

$$\tilde{S}_k = \sum_{m=1}^M S_m \exp\left(i2\pi \left(\frac{k}{M}\right) m\right), \quad (5.21)$$

resulting in a complex sequence $\{\tilde{S}_k\}$, $k \in [1, M]$. After this process, we simply limit the bandwidth by zeroing the relevant frequency components,

$$\tilde{S}_k = \begin{cases} \tilde{S}_k & k \in [1, N_{\max}] \cup [M - N_{\max}, M] \\ 0 & k \in [N_{\max} + 1, M - N_{\max}] \end{cases}. \quad (5.22)$$

Once they are zeroed out, we can take the inverse transformation to transform the pulse back to the time domain,

$$S_k = \frac{1}{M} \sum_{k=1}^M \tilde{S}_k \exp\left(i2\pi \left(\frac{m}{M}\right) k\right), \quad (5.23)$$

effectively low-pass filtering the pulse to stay within the available bandwidth.

5.3.4 Evolution Propagator

To understand how the control pulses evolve our system, we start with the open quantum system master equation for the simulated system,

$$\frac{d}{dt}\rho_S(t) = -i[\mathcal{H}_S(t), \rho_S(t)] + \mathcal{L}(t), \quad (5.24)$$

where we make use of our interpolation objects to find the effective parameters in terms of experimental controls,

$$\begin{aligned}
\mathcal{H}_S(t) = & \underbrace{\sum_{j=1}^N \frac{1}{2} \left[\left(1 + \frac{\delta g_j(\vec{V}(t), \vec{W}(t))}{2} \right) \omega - \omega_{RF} \right] Z_j}_{\text{Zeeman}} + \underbrace{\sum_{j=1}^{N-1} \frac{J_j(\vec{V}(t), \vec{W}(t))}{4\hbar} \vec{\sigma}_j \cdot \vec{\sigma}_{j+1}}_{\text{exchange}} \\
& + \underbrace{\frac{\mu_B B_{RF}(t)}{\hbar} \left[\cos \phi(t) \sum_{j=1}^N X_j + \sin \phi(t) \sum_{j=1}^N Y_j \right]}_{\text{radio frequency field}},
\end{aligned} \tag{5.25}$$

and,

$$\mathcal{L}(t) = \sum_{\alpha=1}^3 \sum_{j=1}^N \gamma_{\alpha,j}^2 \left(L_{\alpha,j} \rho_S(t) L_{\alpha,j}^\dagger - \frac{1}{2} \left\{ L_{\alpha,j}^\dagger L_{\alpha,j}, \rho_S(t) \right\} \right). \tag{5.26}$$

To leverage machine learning, we must convert this equation into a matrix equation. This can be done by using the Choi-Jamiolkowski isomorphism [32],

$$|i\rangle \langle j| \leftrightarrow |j\rangle \otimes |i\rangle. \tag{5.27}$$

As an illustration,

$$\rho = \sum_{i,j} \rho_{i,j} |i\rangle \langle j| \leftrightarrow \text{vec} \rho = \sum_{i,j} \rho_{i,j} |j\rangle \otimes |i\rangle, \tag{5.28}$$

which is equivalent to stacking the columns of the density matrix,

$$\text{vec} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a \\ c \\ b \\ d \end{pmatrix}. \tag{5.29}$$

We can apply this theory to obtain useful properties. Suppose we want to vectorize the matrix product of three matrices ABC , using our isomorphism we can show that it is,

$$\text{vec}(ABC) = (C^T \otimes A) \text{vec}(B), \tag{5.30}$$

and as a relevant example,

$$\text{vec}(\mathcal{H}_S \rho) = \text{vec}(\mathcal{H}_S \rho \mathbb{I}) = (\mathbb{I} \otimes \mathcal{H}_S) \text{vec} \rho. \quad (5.31)$$

Using this isomorphism, the master equation can be written as a large matrix being applied to the density matrix [25]. Taking the master equation term-by-term we first get,

$$\text{vec}(-i[\mathcal{H}_S(t), \rho_S(t)]) \leftrightarrow -i[\mathbb{I} \otimes \mathcal{H}_S(t) - \mathcal{H}_S(t)^T \otimes I] \text{vec} \rho_S(t), \quad (5.32)$$

and then,

$$\text{vec}(\mathcal{L}(t)) \leftrightarrow \sum_{\alpha=1}^3 \sum_{j=1}^N \gamma_{\alpha,j}^2 \left[L_{\alpha,j}^* \otimes L_{\alpha,j} - \frac{1}{2} \mathbb{I} \otimes L_{\alpha,j}^\dagger L_{\alpha,j} - \frac{1}{2} (L_{\alpha,j}^\dagger L_{\alpha,j})^T \otimes \mathbb{I} \right] \text{vec} \rho_S(t). \quad (5.33)$$

Therefore, we can now get a differential matrix master equation,

$$\frac{d}{dt} \text{vec} \rho_S(t) = \mathbb{L}(t) \text{vec} \rho_S(t), \quad (5.34)$$

where,

$$\begin{aligned} \mathbb{L}(t) = & -i[\mathbb{I} \otimes \mathcal{H}_S(t) - \mathcal{H}_S(t)^T \otimes I] \\ & + \sum_{\alpha=1}^3 \sum_{j=1}^N \gamma_{\alpha,j}^2 \left[L_{\alpha,j}^* \otimes L_{\alpha,j} - \frac{1}{2} \mathbb{I} \otimes L_{\alpha,j}^\dagger L_{\alpha,j} - \frac{1}{2} (L_{\alpha,j}^\dagger L_{\alpha,j})^T \otimes \mathbb{I} \right], \end{aligned} \quad (5.35)$$

with solution given by,

$$\text{vec} \rho_S(t) = \mathbb{G}(t) \text{vec} \rho_S(t_0), \quad \mathbb{G}(t) = \mathcal{T} \exp \left(\int_0^t dt \mathbb{L}(t) \right), \quad (5.36)$$

where $\mathbb{G}(t)$ is known as the propagator of the dynamical map $\mathbb{L}(t)$ or in other words the *superoperator*. The density matrix at the end of a pulse sequence would be given by,

$$\text{vec} \rho_S(T) = \mathbb{G}(T) \text{vec} \rho_S(t_0). \quad (5.37)$$

5.3.5 Cost Function

Assuming the desired unitary gate operation is $U_0(t)$, the density matrix evolution can be written as,

$$\text{vec}\rho(t) = \mathbb{U}_0(t)\text{vec}\rho_0, \quad \mathbb{U}_0(t) \equiv U_0^*(t) \otimes U_0(t). \quad (5.38)$$

Comparing the evolution propagator and the desired unitary operator we obtain the unitary gate fidelity,

$$\mathcal{F}_{\text{gate}}(\mathbb{G}(T), \mathbb{U}_0(T)) = \frac{1}{\dim \mathbb{U}_0} |\text{Tr} [\mathbb{G}(T), \mathbb{U}_0(T)]|, \quad (5.39)$$

where T is the length of the pulse [25]. The objective function used for the optimization of this metric is the conventional mean-squared loss,

$$C = (1 - \mathcal{F}_{\text{gate}})^2. \quad (5.40)$$

More constraints and penalties can be added to this cost function as the user desires.

5.3.6 Open System Pulse Design

In order to test the protocol, two fundamental types of unitary operations were chosen to be designed; each of which help to confirm the protocols behaviour. The first is the Hadamard gate that is only applied to the first qubit with the second qubit idling; this requires minimal exchange interaction and delicate g -factor and ESR field deviations. The second gate operation, and arguably the most important, is the $\sqrt{\text{SWAP}}$ gate; a gate that requires carefully evolving the states through the Heisenberg exchange interaction. Both of these circuits are visualized in Figure 5.6

Together, these two gates demonstrate the protocols ability to perform single qubit rotation and entanglement gates which form a complete set for quantum computing in this platform. Any other gates (including the controlled gates) can be constructed from them. Figure 5.7 shows how the Controlled- Z gate can be made. Showing high fidelity gate operations for these two circuits prove the system's ability to design pulses for all the different kinds of unitary gates.

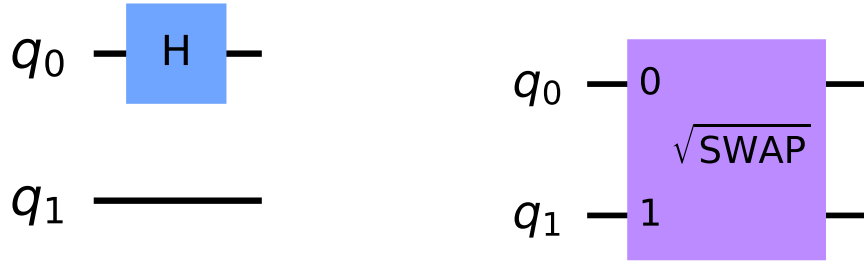


Figure 5.6: The unitary gate operations that was used to test the performance of the pulse design protocol that accomplish (left) single qubit rotations with pre-dominantly g -factor deviations and ESR field and (right) delicate exchange interactions.

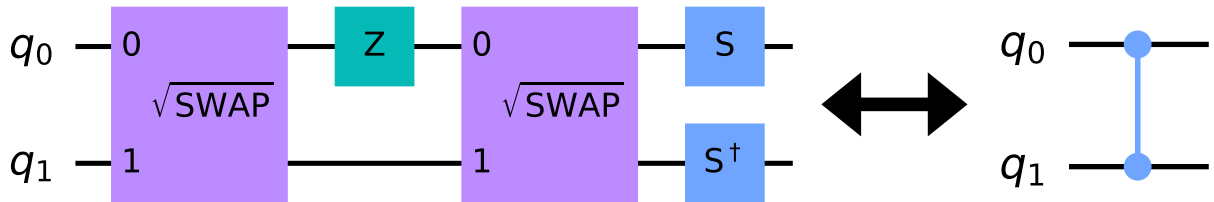


Figure 5.7: Example showing how the controlled-Z gate can be constructed using just single qubit rotations and the $\sqrt{\text{SWAP}}$ gate.

The system parameters used for optimization are shown in Table 5.1. The hyper-parameters used for training are shown in Table 5.2.

System Parameter	Symbol	Value	Unit
Relaxation Time	T_1	28	ms
Dephasing Time	T_2	$\sqrt{32}$	μs
Electron Temperature	T_e	0.1	K
Zeeman Field Strength	B_0	607.285	mT
Bandwidth	$\omega_{\max}$	1	GHz
RF Frequency	ω_{RF}	17	GHz

Table 5.1: System parameters used for the optimization of the Hadamard and $\sqrt{\text{SWAP}}$ gates.

Hyper-Parameter	Symbol	Value
Minimum Fidelity	$\mathcal{F}_{\min}$	0.999
Max Learning Rate	$\eta_{\max}$	10^{-3}
Min Learning Rate	$\eta_{\min}$	10^{-6}
Max Epoch	$n_{\max}$	50000
Max Temperature	$T_{\max}$	1
Learning Rate Scheduler	N/A	$\eta_{\min} + \frac{1}{2}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{T}{T_i}\pi\right)\right)$
Temperature Scheduler	N/A	$\frac{1}{1+\log n}$
Signal Expansion	N/A	Gaussian
Time Step	Δt	1 ns

Table 5.2: Hyper-parameters used for the optimization of the Hadamard and $\sqrt{\text{SWAP}}$ gates.

Hadamard Gate

The first thing to look at is the infidelity curve in Figure 5.8 for the optimization process.

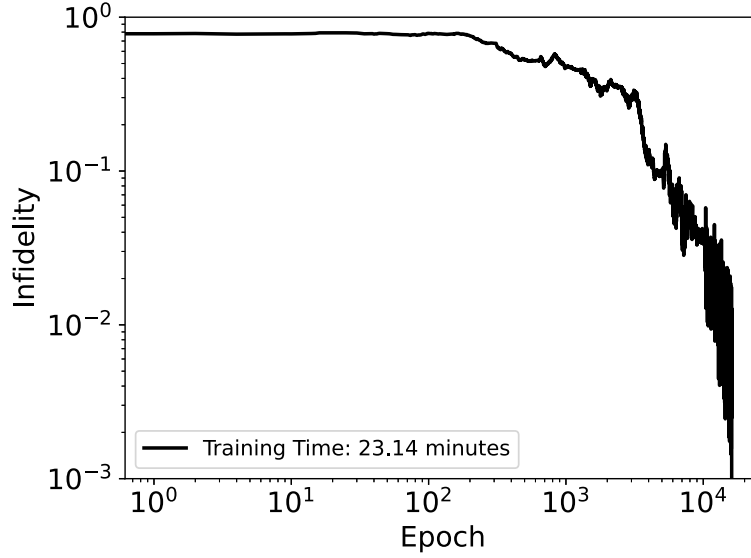


Figure 5.8: Hadamard Gate: The infidelity, $1 - \mathcal{F}$, curve during the training process. It took just over twenty minutes to find a set of experimental signals that realize the gate to a fidelity above the error correction threshold.

After just over twenty minutes, the protocol was able to achieve a set of pulses that achieve a unitary gate infidelity below the error correction threshold. Confident in the optimization, the experimental controls obtained in Figure 5.9 can be studied. This is followed by inspecting the effective control parameters obtained through the interpolation object shown in Figure 5.10. The exchange interaction is essentially zero as expected for purely single qubit rotations. The g -factor for each spin qubit is significantly deviated in order to perform the rotations around the Z -axis needed for a Hadamard gate. Finally, we can visualize the unitary evolution by setting an initial state and evolving it through the propagator. This is shown in Figure 5.11.

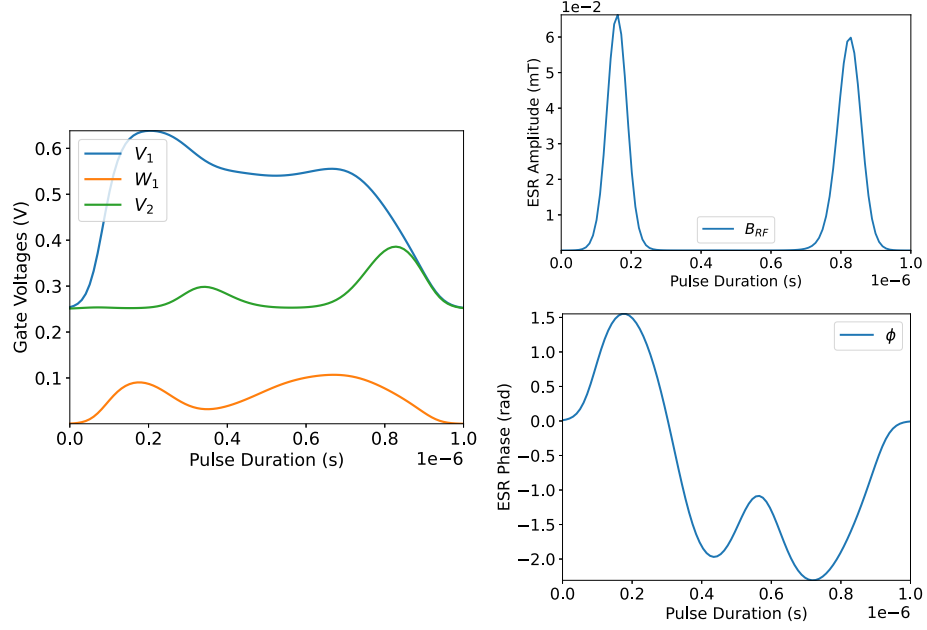


Figure 5.9: Hadamard Gate: (left) The optimized experimental voltage gate pulses. (right) ESR amplitude and phase respectively.

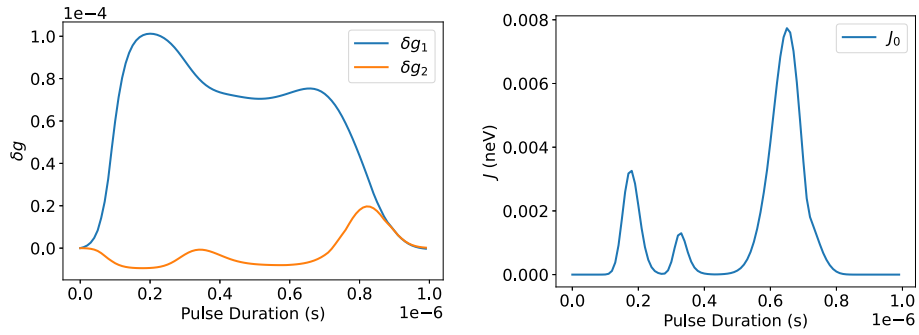


Figure 5.10: Hadamard Gate: (left) Interpolated g -factor deviations for the two qubits and (right) the exchange coupling between the two qubits. As expected for a single qubit rotation little to no exchange interaction is required.

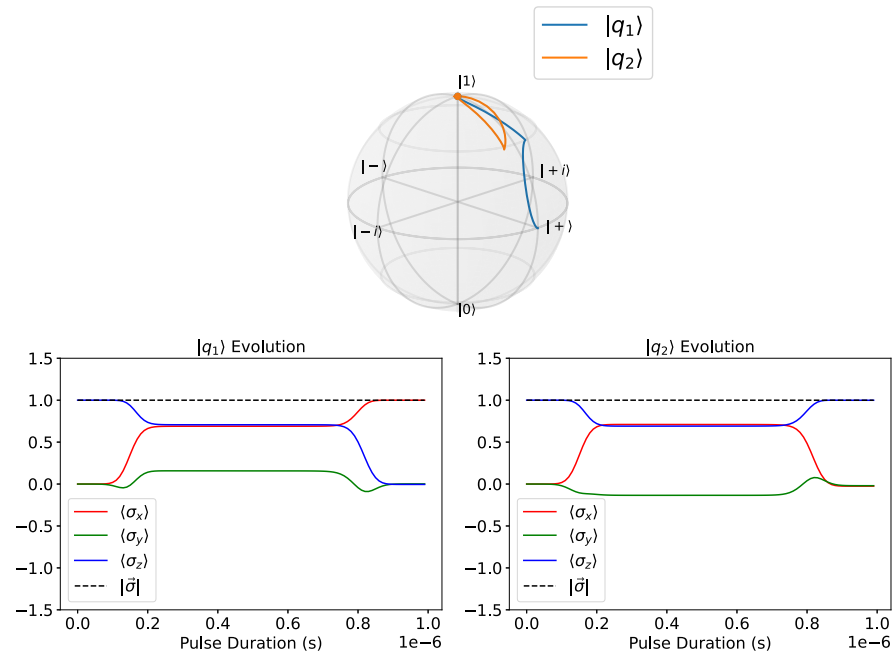


Figure 5.11: Hadamard Gate: State evolution of $|q_1 q_2\rangle = |11\rangle$ with unitary gate infidelity of 10^{-3} . The states remain pure during the evolution as desired.

$\sqrt{\text{SWAP}}$ Gate

The first thing to look at is the infidelity curve in Figure 5.12. After just over a minute, the protocol was able to optimize experimental pulses that realize the operation to an infidelity below the error correction threshold. With the fidelity below the desired threshold we can

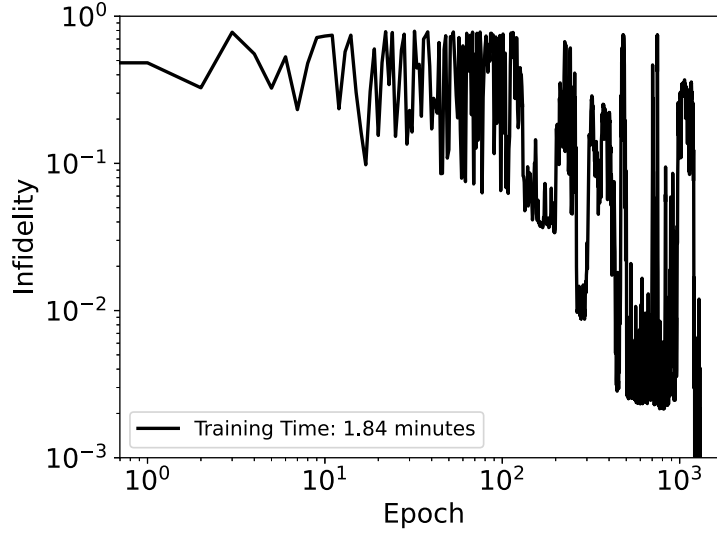


Figure 5.12: $\sqrt{\text{SWAP}}$ Gate: The infidelity, $1 - \mathcal{F}$, curve during the training process. The sensitive nature of the exchange interaction is evident in the oscillatory response of the curve.

take a look at the experimental controls in Figure 5.13. This is followed by inspecting the effective control parameters obtained through interpolation shown in Figure 5.14. As expected, exchange interaction is required with little to no g -factor deviation. Finally, we can plot the unitary evolution by choosing an initial state and evolving it as shown in Figure 5.15.

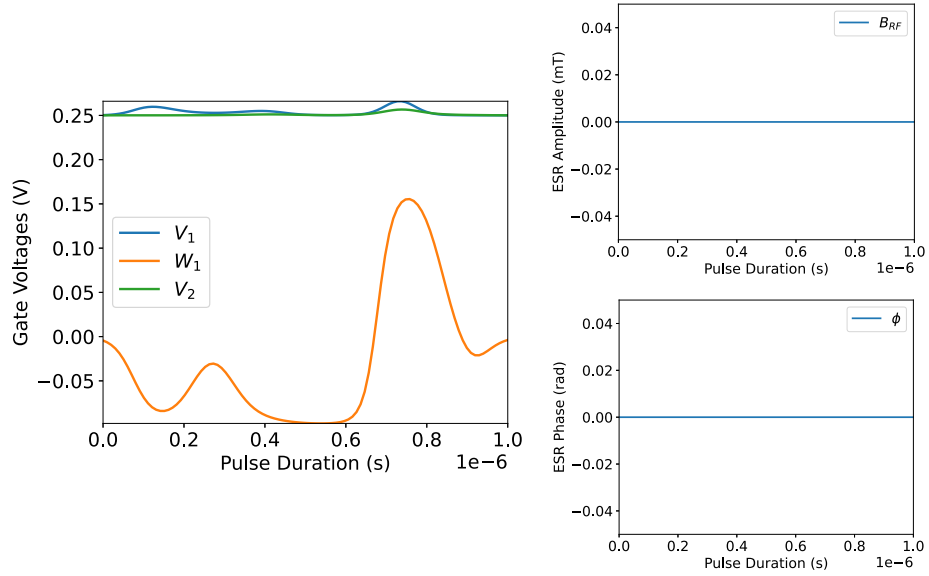


Figure 5.13: $\sqrt{\text{SWAP}}$ Gate: (left) The optimized voltage pulses obtained. (right) The ESR amplitude and phase respectively. Exchange interactions don't require any ESR pulsing so they were set to zero for the duration of the optimization.

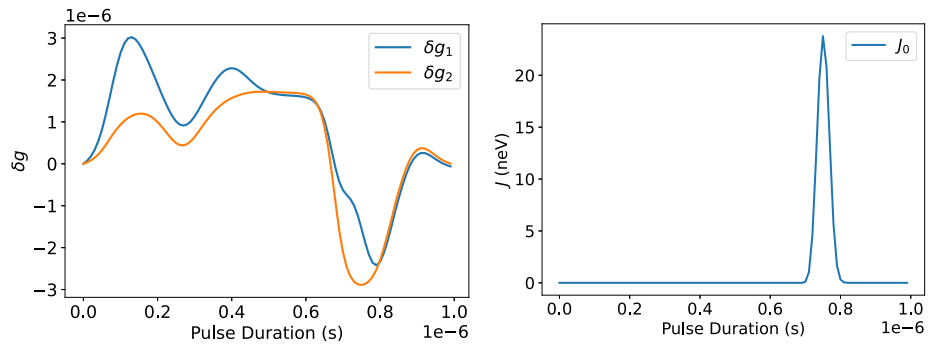


Figure 5.14: $\sqrt{\text{SWAP}}$ Gate: (left) Interpolated g -factor deviations for the two qubits and (right) the exchange coupling between the two qubits. As expected there is very little deviation in the g -factor for the duration of the pulse but a strong exchange interaction for around 100 microseconds.

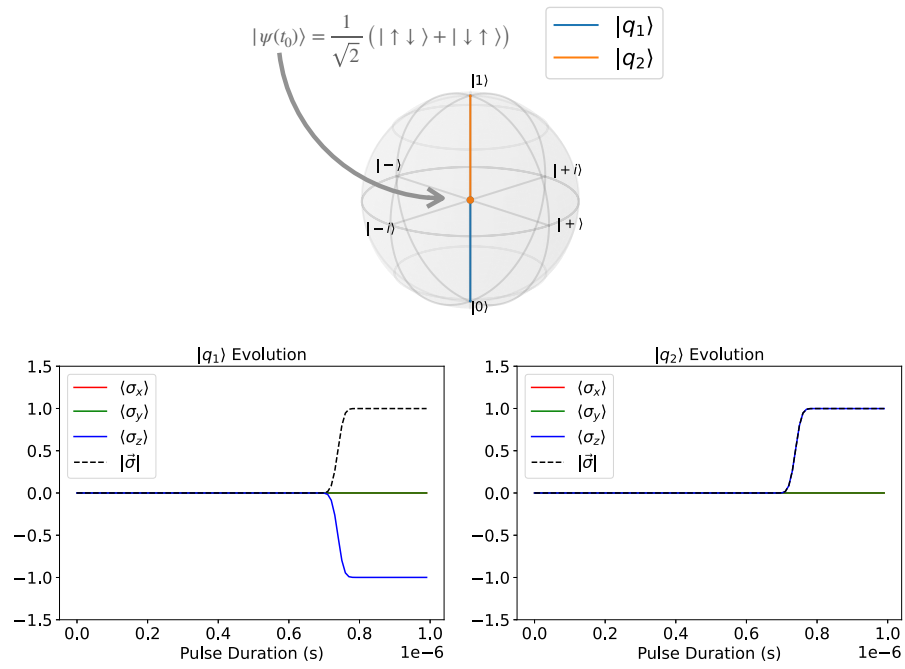


Figure 5.15: $\sqrt{\text{SWAP}}$ Gate: Example initial triplet state evolved under the optimized voltage and ESR pulses of infidelity 10^{-3} . This initial state was chosen to showcase the effect of the unitary gate to “undo” the entanglement to return the expected state at the end of the pulse.

5.4 Experimental Integration

The pulses obtained work very well for the simulated device in QuDiPy. Future work entails taking these optimized experimental pulses and applying them to a real device. Admittedly, these pulses will construct a unitary with fidelity less than what was achieved via the protocol but enhancements can be made to counter this.

5.4.1 Signal Chain Physics

The optimized pulses have to travel from the laboratory equipment to the device (also known as the signal chain). An immediate extension of this work would be to account for losses that the experimental pulses may incur through their travel to the device by measuring and reversing the distortion effect. This has been done with transmon qubits and would be critical to the success of these pulses when applied experimentally to real devices [49]. Very recently (arXiv pre-print), a similar technique was applied to semiconductor silicon spin qubits using the very sensitive exchange interaction as a method of measuring signal chain distortion effects [39].

5.4.2 Randomized Bench-marking

Randomized bench-marking is a method that yields estimates of computationally relevant errors from an experimental quantum system without requiring accurate state preparation and measurement (SPAM) [27]. Remarkably, we can use randomized bench-marking for quantum control in experimental systems by using a subset of measurements to infer fidelity error; this is known as the ORBIT procedure [23]. This technique was applied to superconducting systems and was shown to be effective for improving single and two qubit gate fidelities. Recently, this was embedded into an impressive three stage quantum control protocol [56]. In it, they propose three stages to optimizing control pulses: (1) find pulses that maximize fidelity of a target operation, (2) given the optimized pulse shapes, calibrate their parameters to maximize an experimental figure of merit and (3) optimize model assumptions to reproduce experimental results. The work in this thesis completes the first stage but without devices to run the pulses on, stages two and three were unable to be started.

5.5 Enhancements

Below is a list of enhancements to expand on this work,

- Further investigation into characterizing the performance of the optimizer. This can be done by setting up a batch runs on a high performance computing clusters to gather statistics.
- Further investigation into understanding the impact of hyper-parameters and how they should be set to achieve optimal performance.
- Implement randomized bench-marking as a method of optimizing pulses by interacting with experimental devices.
- Explore performance for other simulated device gate geometries. These gate geometries directly impact the interpolation object mapping voltages to effective parameters.
- Explore gate design for multi dot (3+) systems.

5.6 Summary

In this chapter, the optimal control protocol that designs experimental control pulses to realize a given unitary gate operation was presented. We applied this protocol to design two important gates showing that arbitrary single qubit rotations and entanglement gates can be designed with fidelities above the error correction threshold. The protocol is able to find these pulses reasonably quickly while also optimizing against experimental limitations such as spin decoherence, constrained amplitude and limited bandwidth.

Chapter 6

Conclusions and Outlook

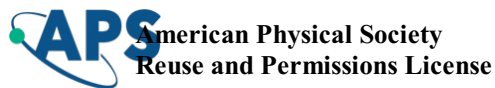
The main focus of this thesis was to address the two main issues that directly affect the scaling of semiconductor quantum dot devices.

In Chapter 4, three major contributions were introduced, each contributed to the overarching goal of automating the tuning process for quantum dot devices. The custom image segmentation software is a novel contribution that leverages Mask-RCNN architecture to address computer vision challenges in quantum dot devices. This was directly followed with a successful, fully automated tuning of a single quantum dot in a n-type, accumulation mode device.

In Chapter 5, an optimal control protocol was presented that is able to optimize experimental control pulses to realize arbitrary quantum gate operations for a simulated two qubit device. The protocol was able to optimize pulses that realize single qubit rotations and entanglement gates to fidelities above the error correction threshold. With these two gates being the fundamental basis gates of spin qubits in semiconductor quantum dots, arbitrary quantum circuits can be constructed using this protocol.

Looking forward, the work presented in this thesis serves as a foundation for future advancements in our research group. The next step would involve fabricating devices, onto which the methodologies and protocols developed in this thesis can be directly applied. With devices at our disposal, we aim to further validate and expand the applicability of these techniques.

Letters of Copyright Permission



13-Sep-2024

This license agreement between the American Physical Society ("APS") and Andrija Paurevic ("You") consists of your license details and the terms and conditions provided by the American Physical Society and SciPris.

Licensed Content Information

License Number:	RNP/24/SEP/083404
License date:	13-Sep-2024
DOI:	10.1103/RevModPhys.95.011006
Title:	Colloquium: Advances in automation of quantum dot devices control
Author:	Justyna P. Zwolak and Jacob M. Taylor
Publication:	Reviews of Modern Physics
Publisher:	American Physical Society
Cost:	USD \$ 0.00

Request Details

Does your reuse require significant modifications:	No
Specify intended distribution locations:	Worldwide
Reuse Category:	Reuse in a thesis/dissertation
Requestor Type:	Student
Items for Reuse:	Figures/Tables
Number of Figure/Tables:	1
Figure/Tables Details:	Figure 2 SEM image of a device
Format for Reuse:	Electronic

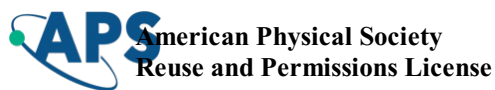
Information about New Publication:

University/Publisher:	University of Waterloo
Title of dissertation/thesis:	Automated Tuning and Optimal Control of Spin Qubits in Quantum Dot Devices
Author(s):	Andrija Paurevic
Expected completion date:	Sep. 2024

License Requestor Information

Name:	Andrija Paurevic
Affiliation:	Individual
Email Id:	apaurevi@uwaterloo.ca
Country:	Canada

Figure 6.1: Reuse and Permissions License for Figure 2.3



13-Sep-2024

This license agreement between the American Physical Society ("APS") and Andrija Paurevic ("You") consists of your license details and the terms and conditions provided by the American Physical Society and SciPris.

Licensed Content Information

License Number:	RNP/24/SEP/083407
License date:	13-Sep-2024
DOI:	10.1103/RevModPhys.79.1217
Title:	Spins in few-electron quantum dots
Author:	R. Hanson et al.
Publication:	Reviews of Modern Physics
Publisher:	American Physical Society
Cost:	USD \$ 0.00

Request Details

Does your reuse require significant modifications:	No
Specify intended distribution locations:	Worldwide
Reuse Category:	Reuse in a thesis/dissertation
Requestor Type:	Student
Items for Reuse:	Figures/Tables
Number of Figure/Tables:	2
Figure/Tables Details:	Figures 3 and Figure 27
Format for Reuse:	Electronic

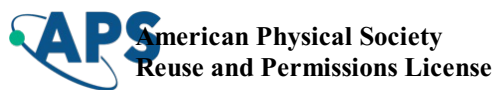
Information about New Publication:

University/Publisher:	University of Waterloo
Title of dissertation/thesis:	Automated Tuning and Optimal Control of Spin Qubits in Quantum Dot Devices
Author(s):	Andrija Paurevic
Expected completion date:	Sep. 2024

License Requestor Information

Name:	Andrija Paurevic
Affiliation:	Individual
Email Id:	apaurevi@uwaterloo.ca
Country:	Canada

Figure 6.2: Reuse and Permissions License for Figures 2.10, 2.6 and 2.5.



13-Sep-2024

This license agreement between the American Physical Society ("APS") and Andrija Paurevic ("You") consists of your license details and the terms and conditions provided by the American Physical Society and SciPris.

Licensed Content Information

License Number:	RNP/24/SEP/083405
License date:	13-Sep-2024
DOI:	10.1103/PhysRevApplied.7.054017
Title:	Microwave Spectroscopy of a Carbon Nanotube Charge Qubit
Author:	Z. V. Penfold-Fitch, F. Sfigakis, and M. R. Buitelaar
Publication:	Physical Review Applied
Publisher:	American Physical Society
Cost:	USD \$ 0.00

Request Details

Does your reuse require significant modifications:	No
Specify intended distribution locations:	Worldwide
Reuse Category:	Reuse in a thesis/dissertation
Requestor Type:	Academic Institution
Items for Reuse:	Figures/Tables
Number of Figure/Tables:	1
Figure/Tables Details:	Figure 5
Format for Reuse:	Electronic

Information about New Publication:

University/Publisher:	University of Waterloo
Title of dissertation/thesis:	Automated Tuning and Optimal Control of Spin Qubits in Quantum Dot Devices
Author(s):	Andrija Paurevic
Expected completion date:	Sep. 2024

License Requestor Information

Name:	Andrija Paurevic
Affiliation:	Individual
Email Id:	apaurevi@uwaterloo.ca
Country:	Canada

Figure 6.3: Reuse and Permissions License for Figure 2.12

References

- [1] Rekha B. S. Object detection using region based convolutional neural network: A survey. *International Journal for Research in Applied Science and Engineering Technology*, 8(7):19271932, July 2020.
- [2] T. Baart, Pieter Eendebak, Christian Reichl, W. Wegscheider, and L. Vandersypen. Computer-automated tuning of semiconductor double quantum dots into the single-electron regime. *Applied Physics Letters*, 108, 03 2016.
- [3] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey, 2018.
- [4] Stefan Birner, Tobias Zibold, Till Andlauer, Tillmann Kubis, Matthias Sabathil, Alex Trellakis, and Peter Vogl. nextnano: General purpose 3-d simulations. *IEEE Transactions on Electron Devices*, 54(9):2137–2142, 2007.
- [5] Heinz-Peter Breuer and Francesco Petruccione. *The Theory of Open Quantum Systems*. Oxford University Press, 01 2007.
- [6] Brandon Buonacorsi, Zhenyu Cai, Eduardo B Ramirez, Kyle S Willick, Sean M Walker, Jiahao Li, Benjamin D Shaw, Xiaosi Xu, Simon C Benjamin, and Jonathan Baugh. Network architecture for a topological quantum computer in silicon. *Quantum Science and Technology*, 4(2):025003, January 2019.
- [7] A. Crippa, Rami Ezzouch, Agostino Apr, A. Amisse, Romain Lavieville, L. Hutin, Benoit Bertrand, M. Vinet, Matias Urdampilleta, C. Buerle, Marc Sanquer, X. Jehl, Romain Maurand, and S. Franceschi. Gate-reflectometry dispersive readout and coherent control of a spin qubit in silicon. *Nature Communications*, 10:2776, 07 2019.
- [8] J. Darulová, S.J. Pauka, N. Wiebe, K.W. Chan, G.C Gardener, M.J. Manfra, M.C. Cassidy, and M. Troyer. Autonomous tuning and charge-state detection of gate-defined quantum dots. *Phys. Rev. Appl.*, 13:054005, May 2020.

- [9] David P. DiVincenzo. The physical implementation of quantum computation. *Fortschritte der Physik*, 48(911):771783, September 2000.
- [10] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):1115, jan 1972.
- [11] R. Durrer, B. Kratochwil, J.V. Koski, A.J. Landig, C. Reichl, W. Wegscheider, T. Ihn, and E. Grepova. Automated tuning of double quantum dots into specific charge states using neural networks. *Phys. Rev. Appl.*, 13:054019, May 2020.
- [12] S.B. Gelfand and S.K. Mitter. Recursive stochastic algorithms for global optimization in ir/sup d/. In *29th IEEE Conference on Decision and Control*, pages 220–221 vol.1, 1990.
- [13] Shekofa Ghoury, Cemil Sungur, and Akif Durdu. Real-time diseases detection of grape and grape leaves using faster r-cnn and ssd mobilenet architectures. 04 2019.
- [14] Ross Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.
- [15] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.
- [16] Coherent Spintronics Group. Qudipy website and documentation. <https://maincsg.github.io/build/html/index.html>, 2024.
- [17] R. Hanson, L. P. Kouwenhoven, J. R. Petta, S. Tarucha, and L. M. K. Vandersypen. Spins in few-electron quantum dots. *Reviews of Modern Physics*, 79(4):12171265, October 2007.
- [18] R. Hanson, L. H. Willems van Beveren, I. T. Vink, J. M. Elzerman, W. J. M. Naber, F. H. L. Koppens, L. P. Kouwenhoven, and L. M. K. Vandersypen. Single-shot readout of electron spin states in a quantum dot using spin-dependent tunnel rates. *Phys. Rev. Lett.*, 94:196802, May 2005.
- [19] Kaiming He, Georgia Gkioxari, Piotr Dollr, and Ross Girshick. Mask r-cnn, 2018.
- [20] Thomas Ihn. Semiconductor nanostructures quantum states and electronic transport. *Semiconductor Nanostructures: Quantum States and Electronic Transport*, 01 2010.

- [21] Hwejin Jung, Bilal Lodhi, and Jaewoo Kang. An automatic nuclei segmentation method based on deep convolutional neural networks for histopathology images. *BMC Biomedical Engineering*, 1(1), October 2019.
- [22] Sandesh Kalantre, Justyna Zwolak, Stephen Ragole, Xingyao Wu, Neil Zimmerman, M. Stewart, and Jacob Taylor. Machine learning techniques for state recognition and auto-tuning in quantum dots. *npj Quantum Information*, 5, 12 2017.
- [23] J. Kelly, R. Barends, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, I.-C. Hoi, E. Jeffrey, A. Megrant, J. Mutus, C. Neill, P. J. J. O’Malley, C. Quintana, P. Roushan, D. Sank, A. Vainsencher, J. Wenner, T. C. White, A. N. Cleland, and John M. Martinis. Optimal quantum control using randomized benchmarking. *Phys. Rev. Lett.*, 112:240504, Jun 2014.
- [24] Navin Khaneja, Timo Reiss, Cindie Kehlet, T. Schulte-Herbrueggen, and Steffen Glaser. Optimal control of coupled spin dynamics: Design of nmr pulse sequences by gradient ascent algorithms. *Journal of magnetic resonance (San Diego, Calif. : 1997)*, 172:296–305, 03 2005.
- [25] Khromets, Bohdan. Heterostructure development and quantum control for semiconductor qubits. Master’s thesis, 2022.
- [26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
- [27] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland. Randomized benchmarking of quantum gates. *Phys. Rev. A*, 77:012307, Jan 2008.
- [28] LP Kouwenhoven, CM Marcus, PL McEuen, S Tarucha, RM Westervelt, and NS Wingreen. Electron transport in quantum dots. In LP Kouwenhoven and GB Schon, editors, *NATO-ASI Workshop on Mesoscopic Electron Transport*, pages 105–214, 1997.
- [29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84 – 90, 2012.
- [30] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

- [31] Ruoyu Li, Luca Petit, David P. Franke, Juan Pablo Dehollain, Jonas Helsen, Mark Steudtner, Nicole K. Thomas, Zachary R. Yoscovits, Kanwal J. Singh, Stephanie Wehner, Lieven M. K. Vandersypen, James S. Clarke, and Menno Veldhorst. A cross-bar network for silicon quantum dot qubits. *Science Advances*, 4(7):eaar3960, 2018.
- [32] Jin-Min Liang, Shi-Jie Wei, and Shao-Ming Fei. Quantum gradient descent algorithms for nonequilibrium steady states and linear algebraic systems. *Science China Physics, Mechanics Astronomy*, 65, 05 2022.
- [33] S. Liles, Ruoyu Li, Henry Yang, Fay Hudson, M. Veldhorst, A. Dzurak, and A. Hamilton. Spin filling and orbital structure of the first six holes in a silicon metal-oxide-semiconductor quantum dot. *Nature Communications*, 9, 08 2018.
- [34] Daniel Loss and David P. DiVincenzo. Quantum computation with quantum dots. *Phys. Rev. A*, 57:120–126, Jan 1998.
- [35] Thomas Walter McJunkin. *Heterostructure modifications, fabrication improvements, and measurement automation of Si/SiGe quantum dots for quantum computation*. PhD thesis, Wisconsin U., Madison, 2021.
- [36] A. Mills, D. Zajac, M. Gullans, Felix Schupp, T. Hazard, and Renato Petta. Shuttling a single charge across a one-dimensional array of silicon quantum dots. *Nature Communications*, 10, 03 2019.
- [37] Filipp Mueller, Georgios Konstantaras, Paul C. Spruijtenburg, Wilfred G. van der Wiel, and Floris A. Zwanenburg. Electronhole confinement symmetry in silicon quantum dots. *Nano Letters*, 15(8):5336–5341, 2015. PMID: 26134900.
- [38] Frederik Nathan and Mark S. Rudner. Universal lindblad equation for open quantum systems. *Phys. Rev. B*, 102:115109, Sep 2020.
- [39] Ming Ni, Rong-Long Ma, Zhen-Zhen Kong, Ning Chu, Wei-Zhu Liao, Sheng-Kai Zhu, Chu Wang, Gang Luo, Di Liu, Gang Cao, Gui-Lei Wang, Hai-Ou Li, and Guo-Ping Guo. Correcting on-chip distortion of control pulses with silicon spin qubits, 2023.
- [40] Jens Hedegaard Nielsen, Mikhail Astafev, William H.P. Nielsen, Dominik Vogel, Iakhotiaharshit, Alex Johnson, AUC Hardal, Akshita, sohail chatoor, Pieter Eendebak, Farzad Bonabi, Sebastian Pauka, Trevor Morgan, Liang, Giulio Ungaretti, Adriaan, Bas Nijholt, qSaavar, Samantha, Pieter Eendebak, Stefan Droege, Jana Darulova, Ruben van Gulik, Natalie Pearson, ThorvaldLarsen, and Andrea Corna. microsoft/q-codes: Qcodes 0.46.0, 2024.

- [41] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.
- [42] Andrija Paurevic. Custom image segmentation. <https://github.com/mainCSG/custom-image-segmentation>, 2024.
- [43] Andrija Paurevic. Diamond miner. <https://github.com/mainCSG/diamondminer>, 2024.
- [44] Andrija Paurevic. Quantum dot control. <https://github.com/mainCSG/QuantumDotControl>, 2024.
- [45] Z. V. Penfold-Fitch, F. Sfigakis, and M. R. Buitelaar. Microwave spectroscopy of a carbon nanotube charge qubit. *Phys. Rev. Appl.*, 7:054017, May 2017.
- [46] Hardik K Prajapati and Rutvij Joshi. Performance analysis of leach with deep learning in wireless sensor networks. *International Journal of Electronics and Telecommunications*, page 799805, November 2022.
- [47] M. Ram, Krithika v r, Priya Batra, and T S Mahesh. Robust quantum control using hybrid pulse engineering. *Physical Review A*, 105, 04 2022.
- [48] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.
- [49] M. Rol, L. Ciorciaro, F. Malinowski, B. Tarasinski, R. Sagastizabal, C. Bultink, Yves Salath, N. Haandbaek, J. Sedivy, and L. DiCarlo. Time-domain characterization and correction of on-chip distortion of control pulses in a quantum processor. *Applied Physics Letters*, 116:054001, 02 2020.
- [50] SciPy. Hough transform. *Hough transform - skimage v0.6 docs*, 2011.
- [51] University of Pittsburgh Sergey Frolov. Quantum Transport Lecture 7. <https://www.youtube.com/watch?v=PXVnvKwn5ak&t=2307s>, 2013.
- [52] Yao Song, Junning Li, Yong-Ju Hai, Qihao Guo, and Xiu-Hao Deng. Optimizing quantum control pulses with complex constraints and few variables through autodifferentiation. *Phys. Rev. A*, 105:012616, Jan 2022.

- [53] Ho Lun Tang, Kyle Connelly, Ada Warren, Fei Zhuang, Sophia E. Economou, and Edwin Barnes. Designing globally time-optimal entangling gates using geometric space curves. *Phys. Rev. Appl.*, 19:044094, Apr 2023.
- [54] W. G. van der Wiel, S. De Franceschi, J. M. Elzerman, T. Fujisawa, S. Tarucha, and L. P. Kouwenhoven. Electron transport through double quantum dots. *Rev. Mod. Phys.*, 75:1–22, Dec 2002.
- [55] Frank K. Wilhelm, Susanna Kirchhoff, Shai Machnes, Nicolas Wittler, Dominique Sugny Theoretical Physics, Saarland University, Saarbrücken, and Laboratoire Interdisciplinaire Carnot de Bourgogne Umr 6303 CN Comt’e. An introduction into optimal control for quantum technologies. *arXiv: Quantum Physics*, 2020.
- [56] Nicolas Wittler, Federico Roy, Kevin Pack, Max Werninghaus, Anurag Saha Roy, Daniel J. Egger, Stefan Filipp, Frank K. Wilhelm, and Shai Machnes. Integrated tool set for control, calibration, and characterization of quantum devices applied to superconducting qubits. *Phys. Rev. Appl.*, 15:034080, Mar 2021.
- [57] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.
- [58] Brian Wuetz, Davide Esposti, A. Zwerver, Sergey Amitonov, Marc Botifoll, Jordi Arbiol, Amir Sammak, Lieven Vandersypen, Maximilian Russ, and Giordano Scappucci. Reducing charge noise in quantum dots by using thin silicon quantum wells. *Nature Communications*, 14, 03 2023.
- [59] Henry Yang, Kok Chan, Robin Harper, Wister Huang, T. Evans, J. Hwang, B. Hensen, Arne Laucht, Tuomo Tanttu, Fay Hudson, S. Flammia, K. Itoh, A. Morello, S. Bartlett, and A. Dzurak. Silicon qubit fidelities approaching incoherent noise limits via pulse engineering. *Nature Electronics*, 2:1, 04 2019.
- [60] Joshua Ziegler, Florian Luthi, Mick Ramsey, Felix Borjans, Guoji Zheng, and Justyna P. Zwolak. Automated extraction of capacitive coupling for quantum dot systems. *Phys. Rev. Appl.*, 19:054077, May 2023.
- [61] Justyna P. Zwolak, Sandesh S. Kalantre, Xian Wu, Stephen Ragole, and Jacob M. Taylor. Qflow lite dataset: A machine-learning approach to the charge states in quantum dot experiments. *PLOS ONE*, 13(10):e0205844, 2018.
- [62] Justyna P. Zwolak and Jacob M. Taylor. Colloquium: Advances in automation of quantum dot devices control. *Rev. Mod. Phys.*, 95:011006, Feb 2023.

- [63] Justyna P Zwolak, Joshua Ziegler, Sandesh S Kalantre, and Jacob M Taylor. QFlow 2.0: Quantum dot data for machine learning, 2022.

APPENDICES

Appendix A

Automated Tuning of Quantum Dot Devices

A.1 Single Quantum Dot Tuner

A.1.1 Bias Point Extraction

Figure [A.1](#) shows how the Hough transform can be used to extract optimal barrier voltage configurations that realize a single central quantum dot.

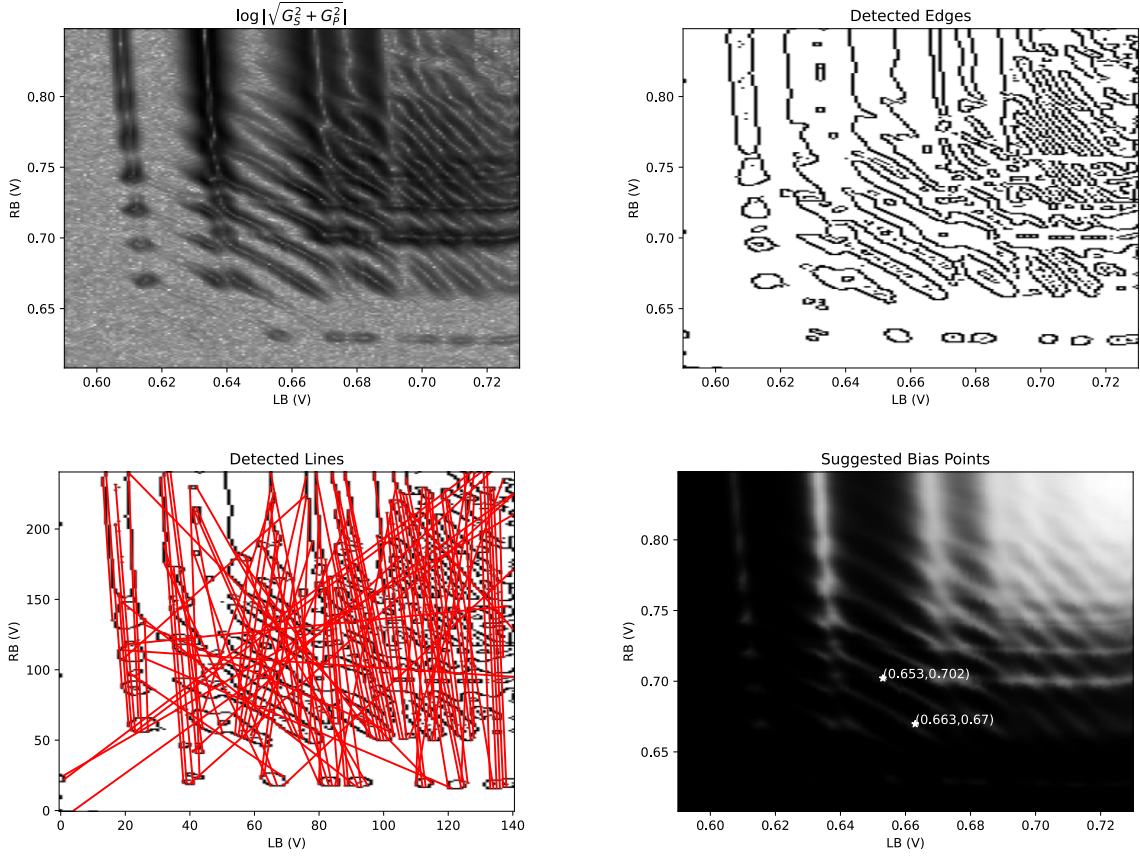


Figure A.1: (top-left) The measurement data is first processed by taking the gradient along a 45 degree line. (top-right) Edges are then extracted and (bottom-left) the lines present in the edges are detected. (bottom-right) Filtering any lines do not meet the criteria of Coulomb oscillations lets us choose the midpoint of said lines at the optimal bias points. In this case, only two bias points were suggested to the user.

A.2 Coulomb Diamond Extraction

Figure A.2 shows how Coulomb diamond measurement data, once thresholded, can be processed into a set of diamonds that can be analyzed.

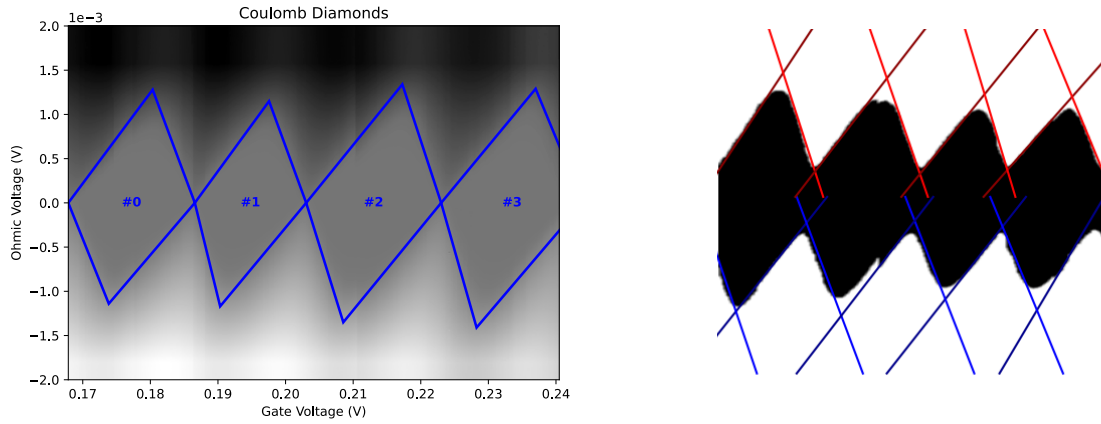


Figure A.2: (left) Raw measurement data with the detected diamonds overlaid on top. (right) By binary thresholding the measurement data (to extract regions of Coulomb blockade), clear diamonds begin to emerge. First, the image is cut in half horizontally and the lines for each half are determined (blue and red). By sorting all detected lines in terms of their half and slope sign, a diamond can be constructed by grouping the appropriate set of lines going from left to right in the image (upper positive + upper negative + lower positive + lower negative = one diamond). The intersection points of all four lines are then found and grouped together to define the geometry of the diamond.