Exploratory Investigation of Surface Code Implementation for Fault-Tolerant Quantum Computing

M. Meulen, BSc, michel.meulen@outlook.com

*From the Master of Applied IT, Fontys University of Applied Sciences, Eindhoven, The Netherlands

(Dated: January 19, 2025)

Quantum computing has the potential to revolutionize computing and have massive effects on major industries such as agriculture, energy, and materials science and more fields by solving computational problems that are intractable with conventional machines. Researchers are making design decisions to attempt to get the most computation out of a machine. One limiting factor is that physical qubits are extremely error prone. This paper aims to explore Quantum Error Correction (QEC) within the context of developing fault-tolerant quantum computers. Specifically, it seeks to enhance QuantumSim, which is a quantum simulator developed by Fontys University of Applied Sciences. This project aims to provide a valuable educational resource by incorporating QEC content into the simulator. The implementation of surface codes, specifically the rotated surface-17 code, is investigated to mitigate the detrimental effects of noise on quantum computations. Building upon foundational concepts such as Shor's nine-qubit code, this study employs a simulation-based approach to compare the performance of the surface code against a non-error-corrected baseline. The simulated noise model, implemented within QuantumSim, incorporates key noise parameters observed in a real quantum device, including depolarizing errors, amplitude damping, and dephasing. While this model provides valuable insights, it does not encompass all potential noise sources encountered in real-world quantum hardware. Future work will focus on incorporating more realistic noise models, including effects such as readout, initialization, and reset errors, to provide a more comprehensive assessment of QEC performance. Despite these limitations, the results obtained with the current noise model demonstrate a significant improvement in computational accuracy when employing the rotated surface-17 code. The surface code exhibits superior resilience to noise across various noise levels, consistently outperforming the non-error-corrected approach. This highlights the crucial role of QEC, particularly surface codes, in enhancing the reliability and scalability of future quantum computing systems.

Keywords: Quantum Error Correction | Shor’s Nine Qubit Code | Rotated Surface-17 Code

1 Introduction

Quantum computing has the potential to revolutionize computing and have massive effects on major industries such as agriculture, energy, and materials science and more fields by solving computational problems that are intractable with conventional machines [1]. As researchers begin to build quantum computing machines of between 50 and 100 quantum bits, also known as qubits, and even larger (e.g. 1000) [1]. They are making design decisions to attempt to get the most computation out of a machine or expand the Simple Quantum Volume (SQV) [1]. SQV can be defined as the number of computational qubits of a machine multiplied by the number of gates expected to be able to perform without error. One limiting factor on SQV as present is that physical qubits are extremely error-prone, which means that computation on these machines is constrained by the short lifetimes of qubits [1] [2]. System designers combat this by attempting to build better physical qubits, but this effort is extremely difficult [3]. Classical systems can be used to alleviate the burden [1]. Specifically, modern Quantum Error Correction (QEC) is a classical control technique and methods that decreases the rate of errors in qubits and expands the SQV [4].

This research aims to explore QEC within the context of developing fault-tolerant quantum computers. Specifically, it seeks to enhance QuantumSim [5], which is a quantum simulator developed by Fontys, primarily used for educational purposes. By incorporating QEC content into the simulator, this project aims to provide a valuable educational resource for students and teachers, enabling them to learn about and teach this critical aspect of quantum computing.

The remainder of this paper is structured as follows: Section 2 contains background information about quantum circuitry, the challenges of QEC, and Shor's nine qubit code [6]. QEC-Codes (QECC) were believed to be impossible till 1995, when Shor demonstrated a 9-qubit QECC which was capable of correcting a single qubit error for the first time [6] [7]. By looking at his work the fundamentals of QEC are explained. Section 2 also highlights a more efficient approach to QECC by using surface codes [4]. Furthermore, this research contains a simulation of the rotated surface-17 code described in Section 3. Both the error corrected, and non-error corrected code are benchmarked to see if applying the surface code concept increases the noise resilience of qubits. These results from the benchmarks are described in Section 4. Section 5 concludes the research, and discusses future work recommendations for the next iteration.

2 Background

This section delves into the foundational principles of QEC. It explores key concepts such as the operation of quantum- circuits and gates, the various challenges encountered in implementing QEC, examines Peter Shor's influential nine-qubit code and concludes with an introduction to the concept of surface codes, a promising approach for building fault-tolerant quantum computers [8].

2.1 Quantum Gates & Circuitry

In quantum computing, quantum circuits are run on a quantum machine. These quantum circuits, consisting of quantum (logic) gates, can be compared to the classical AND, OR, XOR and NOT logic gates [9]. A couple of general quantum gates and their usability regarding QEC will be presented here. There are more generic quantum gates, but are not relevant to the quantum circuits presented later in this document. Starting with the single qubit gates and the Bloch sphere:

Bloch Sphere

Before explaining the different quantum gates. It is important to understand the Bloch sphere. In quantum mechanics and computing, the Bloch sphere is a geometrical representation of the pure state space of a qubit.

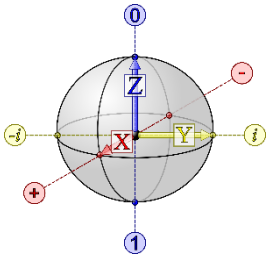

Figure 1 | Visualization of the Bloch Sphere, each capital letter representing their specific axis: x (red), z (blue), y (yellow)

Pauli X, Y, Z gate

The Pauli gates are single qubit gates which rotate the qubit 180 degrees around their specific X, Y, Z axis, which can be visualised using the Bloch Sphere in Figure 1. X is negation (the classical NOT operation on $|0\rangle$ and $|1\rangle$ viewed as classical bits), Z changes the relative phase of a superposition in the standard basis, and Y = ZX is a combination of negation and phase change [9]. In graphical notation, these gates are represented by boxes with a capital letter representing which axis they have an effect on, see Figure 2.

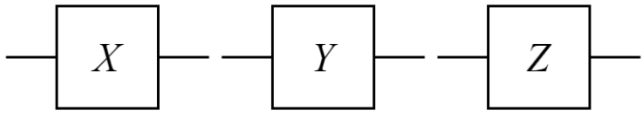

Figure 2 | Graphical notation of the Pauli X, Y and Z-gates

The Pauli X gate is often referred to as the NOT-gate. It also has an alternative notation shown in Figure 3.

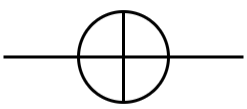

Figure 3 | Alternative graphical notation of the Pauli X (NOT) gate

Hadamard gate

The Hadamard transformation is an important single-qubit transformation which produces an even superposition of |0〉 and |1〉 from either of the standard basis elements [9]. This equal superposition is a key aspect of quantum computing. When a qubit is in a superposition state it is called the Hadamard basis. The Hadamard gate is especially useful in QEC. Since the gate is used to detect phase flips. In the Hadamard basis, phase flips appear as bit flips [9] and can therefore be detected, using a combination of gates. In graphical notation, the Hadamard gate is represented by a box with the letter *H*, see Figure 4.

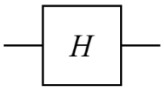

Figure 4 | Graphical notation of the Hadamard gate

CNOT gate

The controlled-not gate, CNOT, controlled Pauli x gate, acts on the standard basis for a two-qubit system, with |0〉 and |1〉 viewed as classical bits, as follows: It flips the second bit, the *target* qubit, if the first bit, the *control* qubit, is 1 and leaves it unchanged otherwise [9].

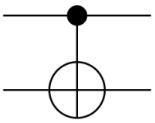

Figure 5 | Graphical notation of the CNOT quantum gate

The CNOT gate is especially useful in quantum computing since it entangles two qubits if the control qubit is in a superposition. Quantum entanglement is a phenomenon where two particles become linked together in such a way that their states are always correlated, regardless of the distance separating them. Examples of quantum entanglement are Bell states.

The notion of the *control* bit and the *target* bit is a carryover from the classical logic gates and should not be taken too literally. In the standard basis, the CNOT operator behaves exactly as the classical gate does on classical bits. However, one should not conclude that the control bit is never changed. When the input qubits are not one of the standard basis elements (0, 1), the effect of the controlled gate can be somewhat counterintuitive. For example, consider the CNOT gate in the Hadamard basis {|+〉 , |−〉 }:

In the Hadamard basis, it is the state of the second qubit that remains unchanged, and the state of the first qubit that is flipped depending on the state of the second bit. Thus, in this basis the sense of which bit is the *control* bit and which the *target* bit has been reversed. But we have not changed the transformation at all, only the way we are thinking about it [9].

Toffoli gate

The Toffoli, controlled CNOT, CCNOT, gate functions the same as the CNOT gate. However, it has one more control qubit than the CNOT gate has. A Pauli x operation is applied on the target qubit if both control qubits are 1 [9]. The Toffoli gate has the following graphical representation:

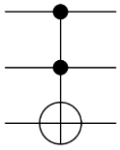

Figure 6 | Graphical notation of the Toffoli gate

Measurement gate

The measurement gate is a single qubit gate and functions as the carry-over from a quantum device to a classical device. The measurement operation destroys entanglement and forces the qubit in the classical state of either |0> or |1>. After measurement the qubit is no longer in a superposition. The measurement gate has the following graphical representation:

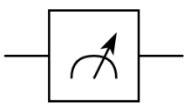

Figure 7 | Graphical notation of the measurement gate

2.2 Challenges of QEC

There are a number of complications that prevent the straight-forward translation of classical codes to quantum codes [10]. The first complication is the no-cloning theorem for quantum states [10], which asserts that it is not possible to construct a unitary operator/ function which duplicates one qubit state to another. In contrast, classical codes work under the assumption that data can be arbitrarily duplicated. For quantum coding, it is therefore necessary to find alternative ways of adding redundancy to the system.

The second complication in quantum coding arises from the fact that qubits are susceptible to both bit-flips (X-errors) and phase-flips (Z-errors) [10]. Quantum error correction codes must therefore be designed with the ability to detect both error-types simultaneously. In contrast, in classical coding, only bit-flip errors need to be considered.

The final complication specific to QEC is the problem of wavefunction collapse [10]. In a classical system, it is possible to measure arbitrary properties of the bit register without the risk of compromising the encoded information. For quantum codes, however, any measurements of the qubits performed as part of the error correction procedure must be carefully chosen so as not to cause the wavefunction to collapse and erase the encoded information [10].

2.3 Peter Shor's Nine Qubit Code

Even though QEC opposes several challenges compared to classical coding. In 1995 Peter Shor published a research paper [6] explaining how to reduce decoherence in quantum computer memory. Many QEC methods refer to his work [1] [7] [8] [10]. Shor showed [6] how to store an arbitrary state of N qubits using nine qubits in a decoherence-resistant way. I.e., even if one of the qubits decoheres, a qubit losing its quantum properties, the original superposition can still be reconstructed perfectly. Shor's nine-qubit code [6] consists of a combination of three; three-qubit bit flip codes and one three-qubit phase flip code [7]. A brief description is provided of the working of these three-qubit codes. For a mapping one to three correction code, 0 is encoded as 000 and 1 is encoded as 111. The redundancies ensure that if a single error has occurred, a majority detector can detect and correct the error, this can be seen in the correction circuit part of Figure 8A.

Due to the no-cloning theorem, we cannot create copies of a certain qubit state [10]. This problem can be solved by encoding the quantum state among three qubits by using two CNOT gates. A single qubit is thus encoded into a 3-qubit state [7]. After encoding, a noisy operation E-bit can be applied, which causes a bit flip on one of the three encoded qubits. The majority detector, as mentioned earlier, is capable of detecting and correcting the error to restore the original superposition.

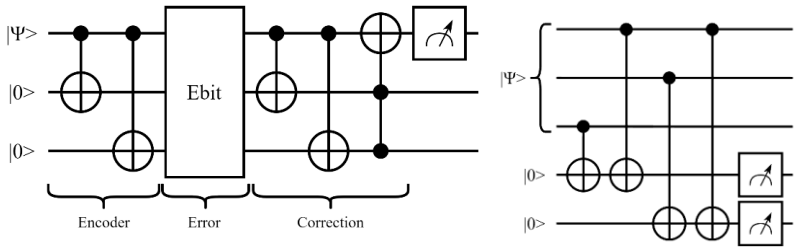

Figure 8 | (A) 3-Qubit bit flip correction circuit | (B) 3-Qubit bit flip syndrome computation circuit

The three-qubit phase flip correction circuit, shown in Figure 9A, looks quite similar to the bit flip correction circuit. However multiple Hadamard gates are added. Rule of thumb is that qubits in the Hadamard basis, turn bit-flips in phase-flips and phase-flips become bit-flips [9]. By putting all redundant/ data qubits in the Hadamard basis, the phase flips can be detected in the correction part of the circuit.

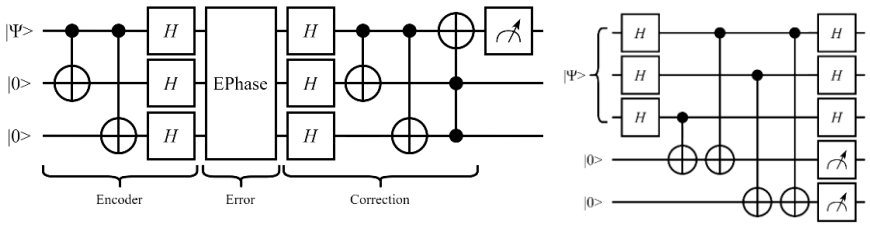

Figure 9 | (A) 3-Qubit bit flip correction circuit | (B) 3-Qubit bit flip syndrome computation circuit

As already mentioned; Shor's nine-qubit code consists of a combination of three qubit bit flip codes, shown in Figure 8A, and one qubit phase flip code, shown in Figure 9A. The Shor nine qubit code can simultaneously protect against up to 1 arbitrary bit-flip error and one arbitrary phase-flip error [11]. The code combines one phase flip correction circuit together with three bit flip correction circuits. A complete overview of the quantum circuit is illustrated in Figure 10, where the fictional Error-gate is the process of generating one possible bit flip and one phase flip. It is crucial to remember that Shor's nine-qubit code relies on an idealized assumption: all gates, except for specific error gates, function perfectly without introducing any errors. This assumption diverges significantly from real-world scenarios where all quantum gates are inherently prone to generating errors. Therefore, Shor's nine qubit code should be primarily viewed as a theoretical implementation rather than a practical example.

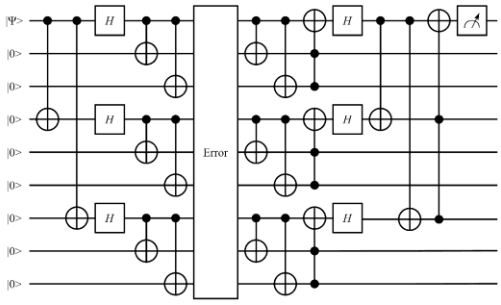

Figure 10 | Shors nine qubit code circuit

The three-qubit- bit flip and phase-flip circuits are perfectly capable of detecting and correcting both type of errors. However, when working with a greater number of qubits this solution is less optimal [10]. QEC codes using more qubits require what is called a decoder. This classical device uses a syndrome to determine the recovery method to remove the error. In order for the decoder to check whether a bit flip error has occurred the syndrome computation/ extraction circuit, shown in Figure 8B & Figure 9B, is used. Such circuits are also used in the surface code, which will be explained in the next section of this paper. The syndrome extraction circuit creates and stores an error syndrome on two redundant, or often called ancillary, qubits. The syndrome contains information if an error occurred and on which data qubit the error is present. If there are no errors the syndrome is measured as 00. If there are errors, both syndrome qubits are not measured as 00. All syndromes possible with the 1/3-bit flip correction circuit are shown in Table 1. Notice that each syndrome is unique, which means additional recovery methods, Pauli operations, can be applied to repair the bit- or phase-flip. Such syndrome extraction methods are also used by the surface code.

Redundant qubits	**Syndrome qubits**
000	00
001	10
010	01
100	11

Table 1 | Bit & phase syndrome computation table, qubits are mapped, the most left qubit in the table being mapped as the top one in the circuit diagram and the most right qubit in the table being mapped to the most right qubit.

2.4 Surface Codes

Surface codes proceed by encoding a set of faulty physical qubits into a set of virtual logical qubits [3]. Information about the current state of the device, called syndromes, are extracted by a specific quantum circuit that does not disturb the underlying computation [10]. Such circuits are shown in Figure 8B & Figure 9B for the 1/3 correction codes. Decoding is the process by which an error correcting protocol maps the measured syndromes information to a set of corrections that, if chosen correctly, should return the system to the correct logical state. A broad overview of the general procedure for executing an error-corrected algorithm with the surface code is shown in Figure 11.

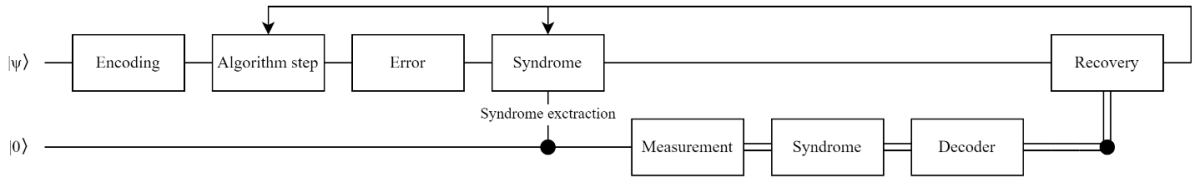

Figure 11 | The general procedure for using a surface code. Double lines indicate classical information flow

The initial step involves encoding the data qubits by entangling them to create a nine qubit logical zero state [12]. Subsequently, the encoded data qubits undergo a single step of the algorithm, which can consist of an arbitrary number of gates implementing the desired computation. Due to the presence of noise within the quantum machine, errors are likely to occur during this algorithmic step. These errors can be detected by measuring stabilizer operators, ensuring that the underlying computation on the data qubits remains unaffected (as will be discussed later). The stabilizer measurement yields an error syndrome. This syndrome information is then transmitted to a classical machine, the decoder, which determines the optimal recovery method. The recovery method involves correcting bit-flip or phase-flip errors on the data qubits. This process can be repeated iteratively, with a trade-off between speed and accuracy.

The realisation of a surface code logical qubit is a key goal for many quantum computing hardware efforts [10]. The general design principle behind topological codes is that the code is built up by 'patching' together repeated elements. We will see that this modular approach ensures that the surface code can be straight-forwardly scaled in size whilst ensuring stabilizer commutativity [10]. In terms of actual implementation, the specific advantage of surface code for current hardware platforms is that it requires only nearest-neighbour interactions. This is advantageous as many quantum computing platforms are unable to perform high-fidelity long-range interactions between qubits [10].

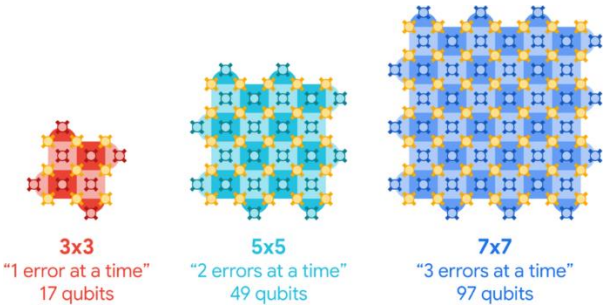

Figure 12 | Surface code logical qubits of increasing sizes, each able to correct more errors than the last. The encoded quantum state is stored on the array of data qubits (gold). Measure qubits (red, cyan, blue) check for errors on the neighboring data qubits [13].

Logical qubits can scale in size, as can be seen in Figure 12, which introduces a new challenge. Making the lattice bigger, using more physical qubits, also introduces more opportunities for error. If the error rate of the physical qubits is too high, these extra errors overwhelm the error correction so that making the lattice bigger just makes the quantum machine's performance worse [13]. Conversely, if the error rate of the physical qubits is sufficiently low, then error correction more than makes up for these extra errors. In fact, the encoded error rate goes down exponentially as more qubits are added [13]. The critical error rate that divides these two cases, below which quantum error correction transforms from harmful to helpful, is called the (pseudo) threshold, shown as the dashed green line in Figure 13. This work from M. Newman et al [13], shows that upscaling the size of the logical qubit lowers the logical error probability, shown in Figure 13. A logical error occurs when the error correction process fails to accurately detect and correct errors in the encoded quantum information.

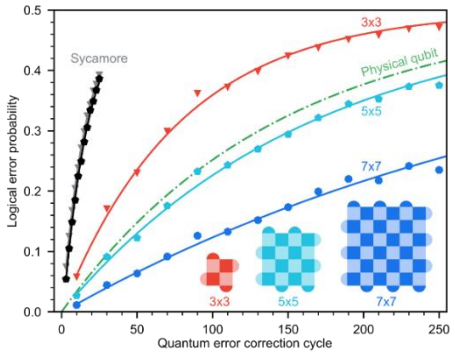

Figure 13 | Logical qubit performance scaling with surface code size. As the lattice grows from 3x3 (red) to 5x5 (cyan) to 7x7 (blue), the logical error probability drops substantially [13].

The surface code is an example of a stabilizer code [10]. The name, *stabilizers*, is based on qubits which are continuously measured during the computation. These special qubits are measured to create error syndromes. These measurements do not affect the logical state of the logical qubit during computation. Properties of different stabilizers codes can be generalised with the [[n, k, d]] stabilizer code notation [10], where *n* is the total number of qubits, *k* is the number of logical qubits and *d* is the code distance. Note, the use of double brackets to differentiate quantum codes from classical codes which are labelled with single brackets. The earlier mentioned Shor's nine qubit code [6] can be notated as [[9, 1, 3]] [10]. The code encodes a qubit state across *nine* different data qubits which represent *one* logical qubit and can sustain a maximum of *two* arbitrary errors before encountering a logical error. A

logical error occurs when the error correction process fails to accurately detect and correct errors in the encoded quantum information. Or in other words an undetectable or uncorrectable error occurs. The minimum number of arbitrary errors to create a logical error is called the code distance [10]. For example, Shor's nine qubit code has a code distance of three due it can detect and correct a maximum of one arbitrary phase- and one arbitrary bit-flip.

Besides having the [[*n, k, d*]] notation there is also the *m* parameter, which represents the number of ancillary qubits used in the surface code [10]. To calculate the number of logical qubits a surface code represents, the equation, shown in Equation 1, is used.

$$\textit{Equation 1}\ [10]\ |\ k = n - m$$

Or in other words:

$$\textit{Equation 2}\ [10]\ |\ number\ of\ logical\ qubits = number\ of\ data\ qubits - number\ of\ ancillary\ qubits$$

For example, the rotated surface-17 code, discussed later, encodes nine physical data qubits, in a three by three lattice, to represent one logical qubit. This means that the code uses: 1 = 9 – m; m = 9 – 1; m = 8. It uses eight ancillary qubits. For the five times five lattice, the surface code uses 25 data qubits to represent one logical qubit. 1 = 25 – m; m = 25 – 1; m = 24. It uses 24 ancillary qubits.

2.5 Stabilizers

While stabilizers have been briefly discussed in previous sections of this paper, their specific operation has not yet been explained. Analogous to the syndrome computation detailed in Section 2.3, stabilizers generate a syndrome that contains information about the detected errors using surface codes. Since quantum error correction requires the detection of both bit-flips and phase-flips, two distinct types of stabilizers are employed: X stabilizers, which check for phase flips and Z stabilizers, which check for bit flips [14]. Stabilizers can either have CNOT or CZ gates as their primitive data-ancilla interaction, as shown in Figure 14 and Figure 15. Within these figures, circles represent the data qubits; squares represent the ancillary qubits. These are special qubits which can be measured during computation without compromising the encoded data of the data qubits. They also differ in the specific order of the interactions with one ancilla qubit. This ensures that all data qubits common to adjacent plaquettes do their inter-actions with one ancilla before the other and provides resilience to ancilla errors in rotated surface-17 codes [14].

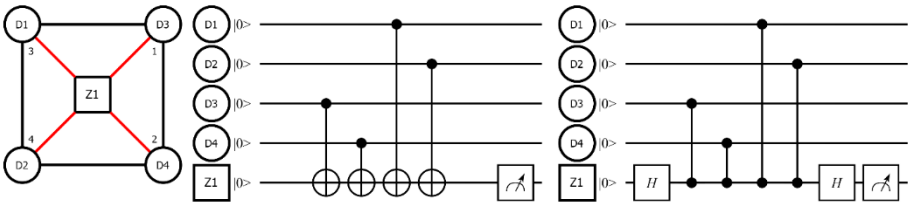

Figure 14 | Visual representation and circuitry of z-type stabilizer, checks for bit-flips

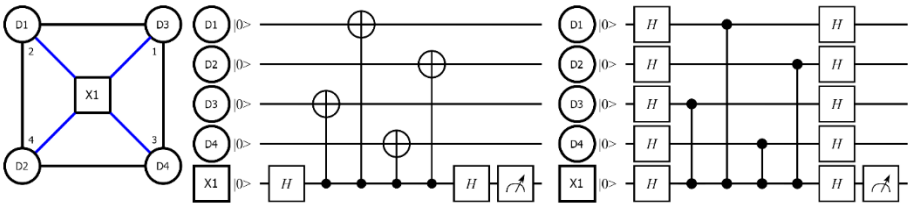

Figure 15 | Visual representation and circuitry of x-type stabilizer, checks for phase-flips

A so-called detection event happens when one of the ancillary qubits detects an error [1]. This happens when one of the ancillary qubits do not commute with all measurements. If there are no errors on all data qubits adjacent to the ancillary qubits, the ancillary qubit is measured as zero. If there is presence of an error on the adjacent qubits the ancillary qubit is measured as one [1]. All ancillary qubits measurements together form the error syndrome. This error syndrome is sent to a classical machine which is called the decoder. This classical device determines, based on the error syndrome, what recovery method should be applied. If the decoder fails in finding a recovery method a logical error occurs. The surface code was not able to repair the error.

3 Methodology

A simulated benchmark was conducted between a non-error-corrected code and an error-corrected code to observe the impact of quantum error correction. The error-corrected code employed the rotated surface-17 code, [[9, 1, 3]]. This code derives its name from the utilization of 17 qubits in its construction [15]. Compared to the surface-25 code, it exhibits reduced qubit and gate requirements and demonstrates superior optimization in terms of execution time compared to the rotated surface-13 code [15]. The surface code uses the layout in illustrated in Figure 16.

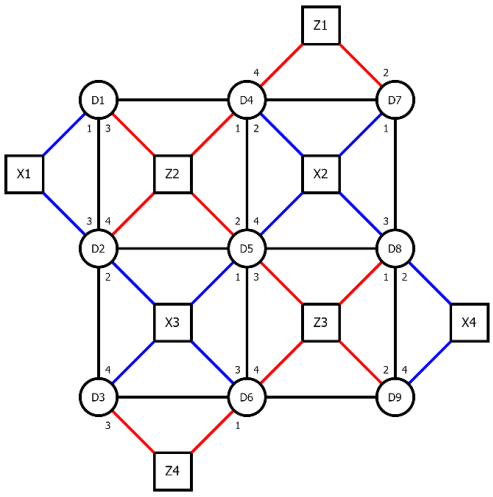

Figure 16 | Visual representation of qubits architecture for the rotated surface-17 code. Circles represent data qubits. Squares represent ancillary qubits used as stabilizers. X-stabilizer interactions are marked blue. Z-stabilizers interactions are marked red. Numbers next to ancillary qubits interactions represent the order of which the ancillary qubits should be executed.

This benchmark evaluates two circuits: one with error correction and one without. The objective is to maintain a state of nine zeros while executing a series of noisy operations. These operations consist of Pauli-X or Pauli-Z gates, applied in sets ranging from 30 to 150 gates with a step size of 30. This arbitrary step size was chosen as smaller increments did not significantly reveal the impact of noise. A maximum of 150 gates was selected to prevent excessively long computation times. The Pauli Z gate is mimicked using two Hadamard gates and a Pauli-X in between. Following the application of these noisy gates, the data qubits are measured, and the number of zeros and ones is recorded. A zero measurement is considered correct, while a one indicates an incorrect answer. Both circuits are subjected to identical noise conditions, as detailed later in this section.

The error-corrected circuit incorporates the rotated surface-17 code. After every series of 30 Pauli gates, the surface code performs stabilizer measurements resulting in an error syndrome. Next, the decoder reads the syndrome and decides the appropriate recovery method, also known as recovery rounds, to remove the error. If the decoder reads an unknown syndrome, which is caused by the presence of more than one error, a logical error is declared, and no further recovery attempts are made for that recovery round. Because the x- and z-stabilizers work independently from each other, at maximum two logical errors per recovery round can occur.

A key distinction lies in the error-corrected circuit. Before computation, all data qubits are encoded into a logical zero state using the nine-qubit surface code, employing the encoder, described in [12] and Figure 17, is a crucial step for surface code operation. Prior to measurement, all data qubits are decoded by applying the inverse of the encoding technique.

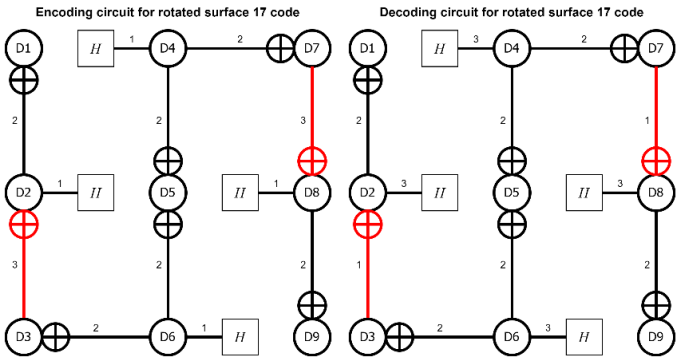

Figure 17 | Encoding (left) and decoding (right) circuitry to create a logical zero state using the nine-qubit surface code [12]. Circles represent data qubits. Each number next to a quantum gate represents the execution order.

All circuits are constructed and simulated using QuantumSim [5]. The Python code for these experiments is available in this repository [5]. The simulator employs a consistent noise model that replicates the noise effects observed in real quantum computers. This noise model, based on the work of G. D. Bartolomeo et al. [16], simulates depolarizing errors, amplitude damping (T1), and dephasing (T2). These noise parameters were extracted from IBM's Kyiv quantum machine. This machine was selected because its noise characteristics were also used to validate and integrate the noise model into QuantumSim [5] by Fontys.

Due to the significant computational cost and memory limitations of QuantumSim, the rotated surface 17 code is implemented using only 10 simulated qubits. This involves a simplification where ancillary qubits are virtually mimicked on a single simulated qubit. Therefore, all qubits share the same noise parameters, whereas in a real quantum device, each qubit exhibits unique noise characteristics.

For this simulation, each qubit is assigned the average p, T1, and T2 values obtained from the Kyiv machine. These values are:

- Qubit's depolarizing error probability chance ratio (p): 0.000277554 → (0.0277554%)
- Qubit's amplitude damping time in nanoseconds (T1): 0.000153157
- Qubit's dephasing time in nanoseconds (T2): 0.000892466

All four circuits were benchmarked across three noise factor scenarios, ranging from one to three. This range was selected to balance the investigation of noise effects while maintaining reasonable computation times. A noise factor of one represents the original averaged noise parameters of the Kyiv machine. Noise factors greater than one amplify the noise, while factors less than one attenuate the noise. The effects on specific p, T1, and T2 values can be calculated using Equation 3, Equation 4 and Equation 5.

$$Equation\ 3 \mid p = p\ from\ Kyiv * noise\ factor$$

$$Equation\ 4 \mid\ T1 = \frac{T1\ from\ Kyiv}{noise\ factor}$$

$$Equation\ 5 \mid\ T2 = \frac{T2\ from\ Kyiv}{noise\ factor}$$

Each benchmark run can produce up to 9 results, corresponding to the nine data qubits in each circuit. Executing the benchmark 100 times generates a maximum of 900 results. To determine correctness, the frequency of "zero" outcomes is analyzed. If a majority of results are zero, they are considered correct. The number of correct answers is calculated by subtracting half of the total runs from the number of zero outcomes, which can be seen in Equation 6.

$$Equation\ 6 \mid Correct\ answers = \ Number\ of\ zero\ outcomes - (\frac{Total\ anwsers}{2})$$

For instance, with 600 zero outcomes and 300 one outcomes (out of 900 total runs), the correct answers are 600 - (900 / 2) = 150. Conversely, with 450 zeros and 450 ones, no majority exists, and the result is zero correct answers, indicating an equal probability of 0 or 1. The 'correct answers' metric serves as a benchmark for evaluating the noise resilience of quantum codes, incorporating both error-corrected and uncorrected implementations. Instead of assessing whether a majority of qubits in a nine-qubit register are zero or one (e.g., 000001111), each qubit measurement is evaluated individually. This approach is more sensitive to the effects of noise. For example, a register with 000001111 would be counted as having four incorrect single-qubit answers, even though it might be considered a "zero" outcome under a different evaluation method.

4 Results

A simulated benchmark was conducted between a non-error-corrected code and an error-corrected code to observe the impact of QEC. The error-corrected code employed the rotated surface-17 code, [[9, 1, 3]]. First the correlation between the noise factor and the number of correct answers is examined for all circuit types. Correct answers are calculated using Equation 6.

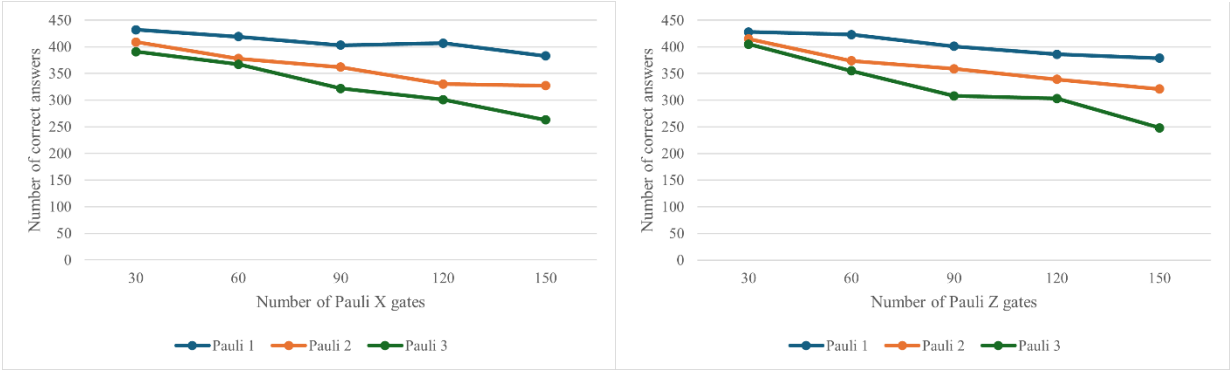

Figure 18 | Effect of Noise Factor on Quantum Circuit Accuracy | Illustrates the impact of increasing noise levels on the accuracy of quantum circuits. The circuits are tasked with maintaining a nine-qubit zero state while subjected to a sequence of Pauli X (left) or Pauli Z (right) gates, ranging from 30 to 150. Performance is evaluated by measuring the number of qubits remaining in the zero state after the Pauli sequence, with a maximum of 450 correct answers achievable. Each line in the figure represents the performance at a different noise factor: 1 (blue), 2 (orange), and 3 (green), where higher factors correspond to increased noise levels.

From Figure 18, it can be observed that the number of correct answers generally decreases with increasing noise, due the noise factor parameter, demonstrating the effect of noise on quantum circuit performance.

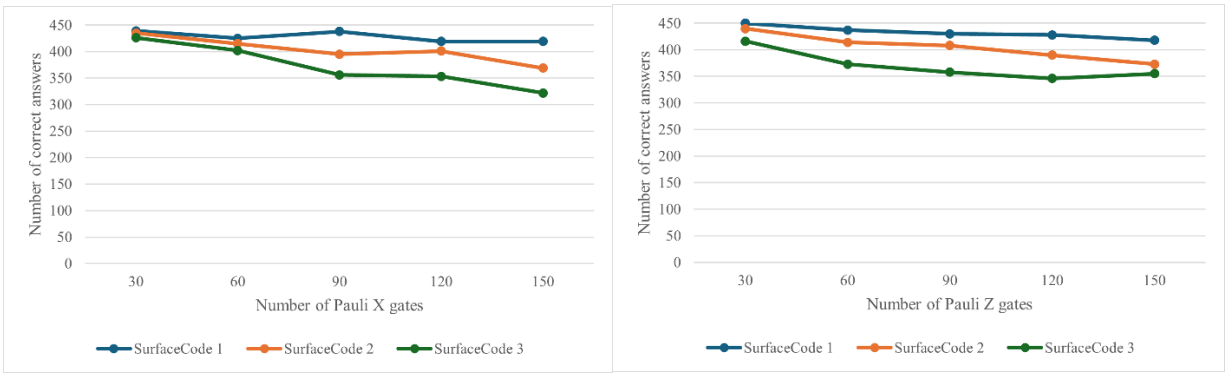

Figure 19 | Noise Resilience of the Rotated Surface-17 Code for Maintaining a Nine-Qubit Zero State | Illustrates the impact of increasing noise, due the noise factor, on the accuracy of quantum circuits employing the rotated surface-17 code, a quantum error-correction (QEC) technique. The circuits are tasked with maintaining a state of nine qubits in the zero state while subjected to a sequence of Pauli X (left) or Pauli Z (right) gates, ranging from 30 to 150. Performance is evaluated by measuring the number of qubits remaining in the zero state after the gate sequence, with 450 representing the maximum possible number of correct answers. Each line represents the performance at a different noise factor: 1 (blue), 2 (orange), and 3 (green), with higher factors indicating greater noise levels. The surface code incorporates a recovery round every 30 Pauli gates to mitigate the effects of noise.

Figure 19 reveals key observations. The number of correct answers generally decreases with increasing noise levels, as indicated by the noise factor parameter, demonstrating the detrimental impact of noise on quantum circuit performance. Figure 18 & Figure 19 highlight that this negative impact of noise is observed across both error-corrected and non-error-corrected quantum circuits, emphasizing the need of noise mitigation in quantum computing.

Figure 20 and Figure 21 visualize a direct comparison between the error-corrected and non-error-corrected circuits.

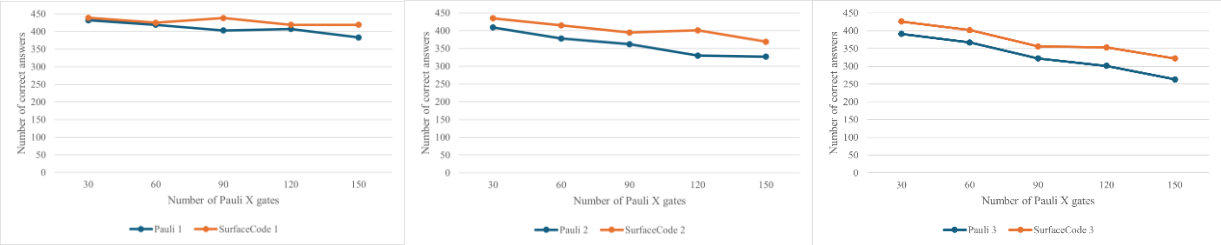

Figure 20 | Pauli X Noise Resilience Comparison: Rotated Surface-17 Code vs. Non-Error-Corrected Code | This figure compares the noise resilience of the rotated surface-17 code (orange) to a non-error-corrected code (blue) under a sequence of Pauli X gates ranging from 30 to 150. Both codes are tasked with maintaining a state of nine qubits in the zero state. The surface-17 code incorporates two recovery rounds. After the Pauli X gate sequence, both codes are measured, and the number of qubits remaining in the zero state is recorded as the number of correct answers, with a maximum of 450. The figure presents results for three different noise factor values: 1 (left), 2 (middle), and 3 (right) with higher factors indicating greater noise levels.

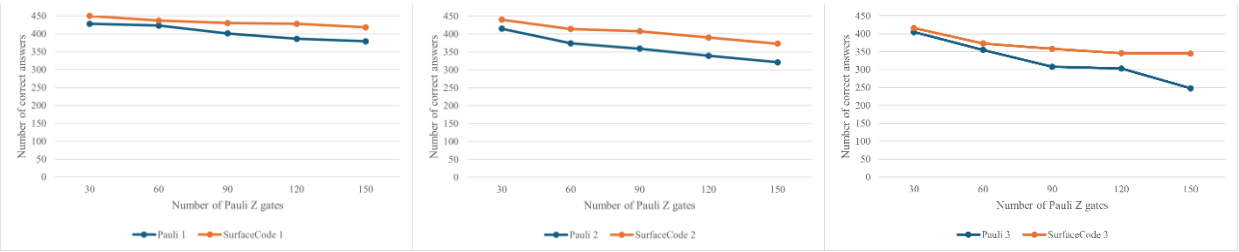

Figure 21 | Pauli Z Noise Resilience Comparison: Rotated Surface-17 Code vs. Non-Error-Corrected Code | This figure compares the noise resilience of the rotated surface-17 code (orange) to a non-error-corrected code (blue) under a sequence of Pauli Z gates ranging from 30 to 150. Both codes are tasked with maintaining a state of nine qubits in the zero state. The surface-17 code incorporates two recovery rounds. After the Pauli Z gate sequence, both codes are measured, and the number of qubits remaining in the zero state is recorded as the number of correct answers, with a maximum of 450. The figure presents results for three different noise factor values: 1 (left), 2 (middle), and 3 (right) with higher factors indicating greater noise levels.

Figure 20 and Figure 21 demonstrate the superior performance of the surface code in maintaining a state with a higher number of zeros, correct answers, compared to the non-error-corrected circuit. Notably, the surface code consistently outperforms the non-error-corrected circuit across all noise factors, from 1 to 3, for both Pauli X and Pauli Z gate sequences.

5 Discussion

This research investigates the fundamental principles of QEC, exemplified by Shor's nine-qubit code and the practical implementation of the rotated surface-17 code. Figure 20 and Figure 21 demonstrate the superior performance of the rotated surface-17 code, a QEC technique, compared to the non-error-corrected circuit. This suggests that QEC could be a step in the right direction towards developing fault-tolerant quantum computers.

The noise model employed in this work, based on the work of G. D. Bartolomeo et al. [16], incorporates several quantum noise parameters, including depolarizing errors (p), amplitude damping (T1), and dephasing (T2). However, it does not encompass all potential noise sources present in real quantum hardware. The current noise implementation lacks consideration for readout, initialization, and reset errors. These error types are likely to have a more significant impact on the error-corrected code due to its increased reliance on measurement and reset gates, which are directly affected by readout and reset errors, respectively. The inclusion of these additional error types could potentially diminish the performance gains observed with QEC techniques like the rotated surface-17 code.

This research does not yet reveal the so-called "pseudo threshold," the point at which the performance of error-corrected and non-error-corrected codes becomes equivalent. This threshold has been demonstrated in other studies, such as [8] and [17], but is not evident in Figure 20 and Figure 21. This discrepancy could be attributed to either the noise factors limited in scope in this study or the absence of certain error types within the current noise model.

Further research is necessary to fully assess the impact of QEC on more realistic noise models that incorporate a broader range of error sources, including readout, initialization, and reset errors. Alternatively, experimental investigations on actual quantum hardware could provide valuable insights into the performance of QEC techniques in the presence of real-world noise.

6 Acknowledgements

We acknowledge the contribution of N. Kuijpers, whose simulator [5] was essential for this research work. Also, we acknowledge the contribution of T. de Laat, whose quantum noise implementation in QuantumSim was essential for this research work.

7 References

[1] A. Holmes, M. R. Jokar, G. Pasandi, Y. Ding, P. Massoud and F. T. Chong, "NISQ+: Boosting quantum computing power by approximating quantum error correction," 14 April 2020. [Online]. Available: https://arxiv.org/pdf/2004.04794. [Accessed 7 October 2024].

[2] R. Harper, S. T. Flammia and a. J. J. Wallman, "Efficient learning of quantum noise," 19 April 2021. [Online]. Available: https://arxiv.org/pdf/1907.13022. [Accessed 29 October 2024].

[3] K. Groenland, "Introduction to Quantum Computing for Business," [Online]. Available: https://quantumcomputingforbusiness.com/. [Accessed 1 October 2024].

[4] Google Quantum AI, "Suppressing quantum errors by scaling a surface code logical qubit," 22 February 2023. [Online]. Available: https://www.nature.com/articles/s41586-022-05434-1. [Accessed 1 October 2024].

[5] N. Kuijpers, "QuantumSim Repository," 2024. [Online]. Available: https://github.com/nicokuijpers/QuantumSim. [Accessed 1 October 2024].

[6] P. W. Shor, "Scheme of reducing decoherence in quantum computer memory," 17 May 1995. [Online]. Available: https://journals.aps.org/pra/abstract/10.1103/PhysRevA.52.R2493. [Accessed 17 October 2024].

[7] A. Mondal and K. P. Keshab, "Quantum Circuits for Stabilizer Error Correcting Codes: A Tutorial," 21 September 2023. [Online]. Available: https://arxiv.org/pdf/2309.11793. [Accessed 28 October 2024].

[8] Google Quantum AI & Collaborators, "Quantum error correction below the surface code threshold," 27 August 2024. [Online]. Available: https://arxiv.org/pdf/2408.13687. [Accessed 1 October 2024].

[9] E. Rieffel and W. Polak, "Quantum Computing a gentle introduction," 2011. [Online]. Available: https://s3.amazonaws.com/arena-attachments/1000401/c8d3f8742d163b7ffd6ae3e4e4e07bf3.pdf. [Accessed 11 November 2024].

[10] J. Roffe, "Quantum Error Correction: An Introductory Guide," 26 July 2019. [Online]. Available: https://arxiv.org/pdf/1907.11157. [Accessed 1 October 2024].

[11] J. Watrous, "Quantum error correction," 21 March 2006. [Online]. Available: https://cs.uwaterloo.ca/~watrous/QC-notes/QC-notes.16.pdf. [Accessed 21 October 2024].

[12] H. Goto, Y. Ho and T. Kanao, "Measurement-free fault-tolerant logical zero-state encoding of the distance-three nine-qubit surface code in a one-dimensional qubit array," 2 June 2023. [Online]. Available: https://arxiv.org/pdf/2303.17211. [Accessed 23 December 2024].

[13] M. Newman and K. Satzinger, "Making quantum error correction work," Google Quantum AI, 9 December 2024. [Online]. Available: https://research.google/blog/making-quantum-error-correction-work/. [Accessed 10 December 2024].

[14] R. Versluis, S. Poletto, N. Khammassi, N. Haider, D. J. Michalak, A. Bruno, K. Bertels and L. DiCarlo, "Scalable quantum circuit and control for a superconducting surface code," 28 December 2016. [Online]. Available: https://arxiv.org/pdf/1612.08208. [Accessed 9 December 2024].

[15] Y. Tomita and K. M. Svore, "Low-distance Surface Codes under Realistic Quantum Noise," 14 April 2014. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/1404.3747v1.pdf. [Accessed 16 December 2024].

[16] G. D. Bartolomeo, M. Vischi, R. Wixinger, M. Grossi, F. Cesa, S. Donadi and A. Bassi, "Noisy gates for simulating quantum computers," 3 February 2023. [Online]. Available: https://journals.aps.org/prresearch/pdf/10.1103/PhysRevResearch.5.043210. [Accessed 16 December 2024].

[17] A. d. iOlius, J. E. Martinez, P. Fuentes, P. M. Crespo and J. Garcia-Frias, "Performance of surface codes in realistic quantum hardware," 21 December 2022. [Online]. Available: https://link.aps.org/accepted/10.1103/PhysRevA.106.062428. [Accessed 25 August 2024].