Surface code quantum computation

Austin G. Fowler
Google Quantum AI
(Dated: December 3, 2024)

We provide a comprehensive introduction to surface code quantum computation. We briefly review quantum states, circuits, Pauli matrices and the Clifford group, then cover stabilizers, the surface code, detectors, surface code initialization, measurement, memory experiments, real-time decoding, identity with movement, correlation surfaces, byproduct operators, CNOT, stabilizer flows, junctions and corners, ZX-calculus, Hadamard and CZ, patch rotation, Y-basis measurement and initialization, the S-gate, state cultivation, the T-gate, Toffoli, arbitrary single qubit unitaries, walking codes, yoking, reaction limited and Clifford-free computation, a detailed study of the overhead of addition, and general techniques to estimate the overhead of algorithms. We finish with areas needing further research.

In 2012 we reviewed surface code quantum computation [1], but a great deal has changed since then. This article covers all the latest results, which inevitably means it is directed at more advanced readers. The reader completely new to the surface code should still start with the older review.

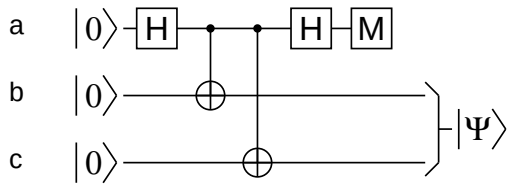

FIG. 1. 3 qubits a, b, c manipulated by a small quantum circuit with measurement-dependent 2-qubit output $|\Psi\rangle$. The two CNOTs effectively implement a controlled-X⊗X gate.

I. PRELIMINARIES

The idea of quantum computation dates back to 1980, with both Yuri Manin [2] and Paul Benioff [3] publishing initial discussions. In 1982, Richard Feynman was the first to speculate that a quantum computer might be able to simulate quantum systems more efficiently than a classical computer [4]. David Deutsch introduced the computational basis $|0\rangle$, $|1\rangle$ in 1985 [5], and we will begin our discussion with the definitions of some common quantum states.

$$|+\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$$
$$|-\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right)$$
$$|T\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{i\pi/4}|1\rangle\right)$$

Quantum circuit diagrams were introduced in 1995 [6], and [7] contains a concise review and the gate definitions we will use. Fig. 1 contains a small example that contains many of the ideas we will need. Let's calculate the output $|\Psi\rangle$.

$$\begin{aligned}
|000\rangle &\xrightarrow{H_a} |+00\rangle \\
&\xrightarrow{CNOT_{ab}} \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right)|0\rangle \\
&\xrightarrow{CNOT_{ac}} \frac{1}{\sqrt{2}}\left(|000\rangle + |111\rangle\right) \\
&\xrightarrow{H_a} \frac{1}{\sqrt{2}}\left(|+00\rangle + |-11\rangle\right) \\
&= \frac{1}{2}|0\rangle\left(|00\rangle + |11\rangle\right) + \frac{1}{2}|1\rangle\left(|00\rangle - |11\rangle\right) \\
&\xrightarrow{M_a} |\Psi\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + (-1)^{M_a}|11\rangle\right)
\end{aligned}$$

In this article, we will spend a lot of time talking about how various combinations of the Pauli matrices $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$, $Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ [8] commute through gates. For example $HX = HXHH = ZH$. Indeed, rather than thinking of H as mapping quantum states in a particular way, it is equivalent to think of the gate as a map of Pauli matrices.

$$X \xrightarrow{H} Z$$
$$Z \xrightarrow{H} X$$

Not all quantum gates can be expressed as mapping Pauli matrices to Pauli matrices. Those that can form a

group called the Clifford group comprised of all quantum circuits consisting of H, S, and $CNOT$ [9]. For a multi-qubit Clifford gate, it is sufficient to specify the action of the gate on X and Z on each input. We will write tensor products of matrices [10] $X_a \otimes I_b$ simply as $X_a I_b$ when the meaning is clear from context.

$$\begin{aligned} X_a I_b &\xrightarrow{CNOT_{ab}} X_a X_b \\ Z_a I_b &\xrightarrow{CNOT_{ab}} Z_a I_b \\ I_a X_b &\xrightarrow{CNOT_{ab}} I_a X_b \\ I_a Z_b &\xrightarrow{CNOT_{ab}} Z_a Z_b \end{aligned}$$

II. STABILIZERS

The concept of a stabilizer group in quantum computing was introduced by Daniel Gottesmann in 1997 [11]. For our purposes, it will not be necessary to discuss stabilizer groups, rather just individual stabilizers, which are just signed tensor products of Pauli matrices that leave a quantum state unchanged.

$$\begin{aligned} Z|0\rangle &= |0\rangle \\ -Z|1\rangle &= |1\rangle \\ X|+\rangle &= |+\rangle \\ -X|-\rangle &= |-\rangle \\ X \otimes X \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\ Z \otimes Z \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \end{aligned}$$

The sign of a stabilizer is very useful for detecting errors in a state. Let A be a stabilizer of $|\Psi\rangle$, so $A|\Psi\rangle = |\Psi\rangle$. Let E be any error that anticommutes with A, so $AE = -EA$. Then if E happens to Ψ we have $E\Psi = EA\Psi = -AE\Psi$. The new state with the error, $E\Psi$, is stabilized by $-A$. If we can somehow track the sign of A, we can determine when errors E occur. Fortunately, there is a very simple circuit for determining the sign of a stabilizer, see Fig. 2.

$$\begin{aligned} |0\rangle|\Psi\rangle &\xrightarrow{H} |+\rangle|\Psi\rangle \\ &\xrightarrow{ctrl-A} \frac{1}{\sqrt{2}}(|0\rangle|\Psi\rangle + |1\rangle A|\Psi\rangle)|0\rangle \\ &\xrightarrow{H} \frac{1}{\sqrt{2}}(|+\rangle|\Psi\rangle + |-\rangle A|\Psi\rangle)|0\rangle \\ &= \frac{1}{2}|0\rangle(|\Psi\rangle + A|\Psi\rangle) + \frac{1}{2}|1\rangle(|\Psi\rangle - A|\Psi\rangle) \\ &\xrightarrow{M} \frac{1}{\sqrt{2}}\left(|\Psi\rangle + (-1)^M A|\Psi\rangle\right) \\ &= |\Psi_M\rangle \end{aligned}$$

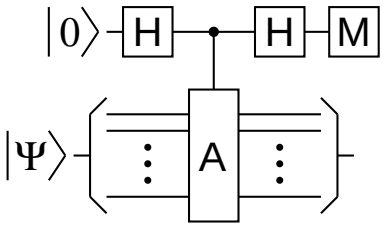

FIG. 2. Given an operator A satisfying $A^2 = I$, this circuit will produce a $+1$ eigenstate of A if $M = 0$, or a -1 eigenstate of A if $M = 1$.

Note that $A|\Psi_M\rangle = (-1)^M|\Psi_M\rangle$. The output of this circuit, independent of what $|\Psi\rangle$ is, will be either the $+1$ or -1 eigenstate of A, heralded by M. Not only does this circuit tell us the sign of A, it cleans up the input as well. An arbitrary computational basis error E will not typically commute or anti-commute with a stabilizer A, but this circuit projects any error to either an error that does, or no error. As an exercise, consider $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, $E = TT$, and $A = XX$. You should be able to show that Fig. 2 produces an error-free state if $M = 0$, or a state with a Z error on one of the two qubits if $M = 1$. If you have trouble showing this, please see Appendix A.

Errors that take the computer out of the computational basis, such as leakage to higher states $|2+\rangle$, must be handled in hardware [12]. Widespread bursts of errors must also be handled in hardware [13]. For the remainder of this article, we shall assume that both non-computational basis and burst errors have been handled in hardware, allowing us to focus on the remaining local, random, independent computational basis errors, which thanks to Fig. 2 we can think of as local, random, independent Pauli errors.

Another great utility of stabilizers lies in their ability to concisely describe the quantum states used for quantum error correction. For example, the state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ is stabilized by XX and ZZ, and is the unique state up to global phase stabilized by these operators. It is therefore appropriate to focus on the list of stabilizers, and cease to write down the state. Given a list of commuting X and Z stabilizers, an explicit state can always be constructed by starting with the all $|0\rangle$ state, which is the simultaneous $+1$ eigenstate of all Z stabilizers, and then one by one making the state also the $+1$ eigenstate of each X stabilizer A via $|\Psi'\rangle = \frac{1}{\sqrt{2}}(|\Psi\rangle + A|\Psi\rangle)$. Note, however, that given a set of independent commuting X and Z stabilizers containing n X stabilizers, the number of computational basis states in the quantum state so constructed will be 2^n, strong motivation to avoid writing down the state.

III. THE SURFACE CODE

Why do we need quantum error correction at all? Why choose the surface code? Instances of commercially or scientifically relevant quantum algorithms likely to outperform classical computers typically consist of at least millions to billions of gates [14]. Reliably executing algorithms at this scale requires a fault-tolerant quantum computation protocol. Given hardware with arbitrary interactions, protocols with constant qubit overhead per error corrected logical qubit do exist [15, 16]. In this article, we focus on 2D arrays of qubits with nearest neighbor interactions only, and for such architectures the leading protocol is surface code quantum computation [17–20]. Significant experimental progress has been made on such architectures, including a real-time-decoded 105 qubit array with a lower memory logical error rate than its best physical qubit, and strong suppression of error as the size of the surface code increases [21].

Before we discuss the surface code, for completeness let's define some common terminology. A state expressed in the Z-basis means a state expressed using the eigenstates of Z, namely $|0\rangle$ and $|1\rangle$, for example $|\Psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. States can also be expressed in the X-basis, meaning using the states $|+\rangle$ and $|-\rangle$, so in the X-basis $|\Psi\rangle = |+\rangle$. Z-basis initialization means initialization to $|0\rangle$, X-basis initialization means initialization to $|+\rangle$. Z-basis measurement M_Z means projection of the quantum state onto the Z-basis, so measuring $|\Psi\rangle$ in the Z-basis will yield the result 0 (+1 eigenstate $|0\rangle$) or 1 (-1 eigenstate $|1\rangle$), each with equal 50% probability. X-basis measurement M_X means projection of the quantum state onto the X-basis, so measuring $|\Psi\rangle$ in the X-basis will yield the result 0 (+1 eigenstate $|+\rangle$) with 100% probability. In most hardware platforms, M_X is achieved by first applying H then M_Z. If in discussion we omit specifying the basis of initialization or measurement, the Z basis is implied.

Consider Fig. 3. This defines colored plaquettes that will be used as compact representations of specific quantum circuits. In this article, we will use an RGB=XYZ mnemonic [22], so the plaquette representing X stabilizer measurement will be red, and that representing Z stabilizer measurement will be blue.

See Fig. 4 for a visual definition of the surface code as a particular scalable quantum circuit with particular logical X and Z operators. The result of any attempt to prepare a +1 eigenstate of logical Z that would work in the absence of errors will be denoted $|0_L\rangle$. For example, suppose every data qubit is initialized to $|0\rangle$. In the absence of errors this is a +1 eigenstate of logical Z. Note, however, that if each data qubit initialization has a probability p of erroneously preparing $|1\rangle$, then for a sufficiently large surface code, the probability that the +1 eigenstate is actually prepared asymptotes to 50%. To understand how to make initialization fault-tolerant, we need to first discuss detectors.

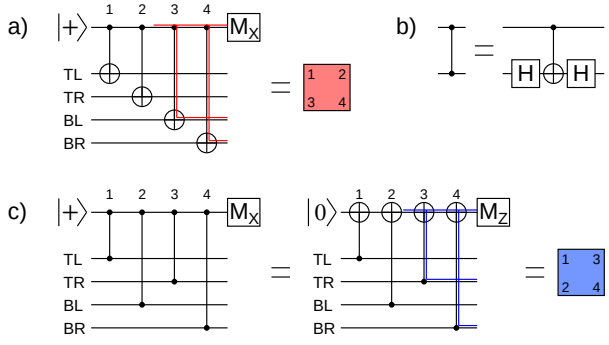

FIG. 3. a) Circuit to measure a 4-qubit X stabilizer. The top qubit is called the measure qubit, the other four are called data qubits. In the visual representation, data qubits are located at the corners, and the measure qubit is associated with the plaquette itself. Numbers near corners show the time step during which the measure and that data qubit interact. Pairs of letters indicate the location of each data qubit in the plaquette. TL: top left, TR: top right, BL: bottom left, BR: bottom right. Red lines show where an X error can occur and propagate to two data qubits. b) A useful quantum circuit identity. c) Two equivalent circuits to measure a 4-qubit Z stabilizer. The right circuit is the one we will make the visual representation equal to. Blue lines show where a Z error can occur and propagate to two data qubits.

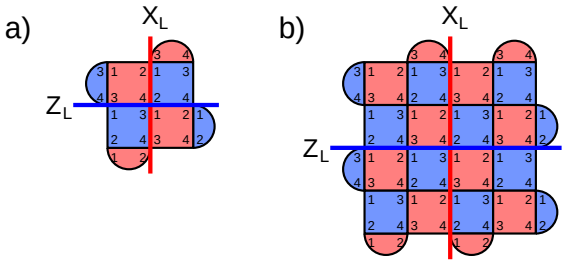

FIG. 4. a) A 3x3 grid of data qubits and 8 measure qubits executing a particular quantum circuit. This is a distance $d = 3$ surface code. Our default choice of logical operators will be chains of physical operators on the horizontal and vertical midlines. b) A $d = 5$ surface code. While it is perfectly possible to define even distance surface codes, we will focus exclusively on odd distance codes in this article.

IV. DETECTORS

The formal definition of a detector is a set of measurements with predictable parity in the absence of errors. The simplest example of a detector is a single qubit initialized to binary state $|b\rangle$ followed by M_Z. In the absence of errors, M_Z and also the parity of the set containing just M_Z will have the predictable value b. Note that

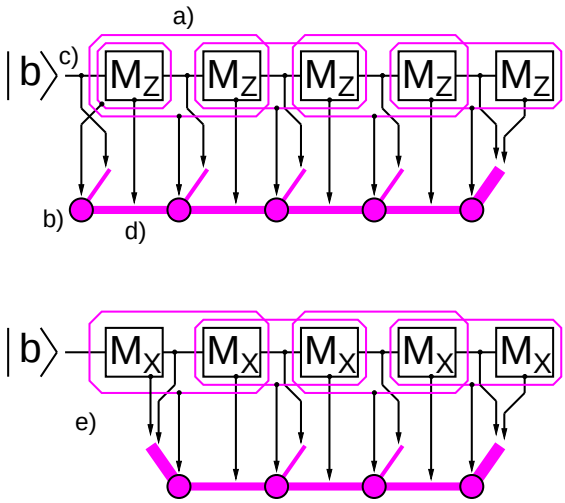

FIG. 5. Repeatedly measuring a binary state $|b\rangle$ in different bases. a) Each magenta boundary encloses a set of measurements that forms a detector, meaning the parity of that set of measurements is known in the absence of errors. b) Each detector can be represented as a node in a graph. The graph consisting of detectors and edges corresponding to error processes is called a detector graph. c) An X error before the first measurement will flip the state and turn just the first detector into a detection event. The probability p of this happening can be represented as a graph edge leading to a special boundary node (not shown). The edge diameter is proportional to p. d) Let a measurement error mean reporting a different state to the one input and output. When a measurement belongs to two detectors, a single measurement error generates two detection events which get connected by an edge. The larger diameter means measurement errors are more likely than errors between measurements in this example. e) When repeatedly measuring M_X, the first measurement is random so errors before it are undetectable. Those during and after this measurement all lead to a detection event on the first detector, collectively one of the most probable error processes in the system.

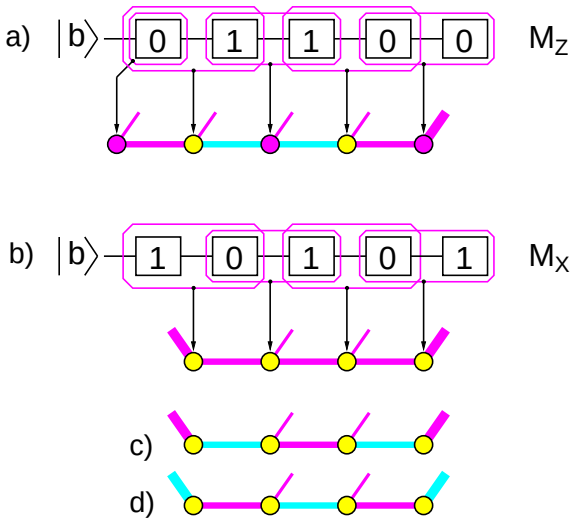

FIG. 6. a) Repeated M_Z experiment from Fig. 5 with specific measurement outcomes. In the absence of errors, the parity of each magenta encircled set of measurements should be even. When this is the case, the associated detector is colored magenta. When the set of measurements has odd parity, the associated detector is colored yellow (indicating a detection event). These detection events can be explained by two X errors, one after the first measurement and one after third measurement, or the second and third measurements reporting the wrong values. The latter case, highlighted in cyan, is more likely since measurement errors have higher probability (represented by thicker edges). A highlighted (cyan) edge can be thought of as toggling the presence of detection events at its endpoints. To explain the observed detection events, a set of edges must be highlighted cyan such that each detection event is incident on an odd number of highlighted edges. The goal is to choose a set of edges with maximum probability. b) Repeated M_X experiment with many errors and detection events. c) One possible explanation of the observed detection events involving two measurement errors. d) Another possible explanation involving one central measurement error and errors on or near the first and last measurements. If these end errors are sufficiently likely, it is possible that both of them occurring has the same probability as a single measurement error elsewhere making this configuration of errors equally likely. This example shows that when edges have different probabilities, the chosen correction may not always involve the minimum number of edges.

the definition of a detector does not require the predicted parity to be 0.

By contrast, if $|b\rangle$ is followed by M_X, then since $|b\rangle = \frac{1}{\sqrt{2}}\left(|+\rangle + (-1)^b |-\rangle\right)$ the result of M_X will be 50/50 random so the set containing just this M_X is not a detector. If we assume measurement is non-demolition, meaning it leaves the qubit in the measured state, a second measurement in the same basis should yield the same result as the first. Indeed, if same basis measurements are repeated indefinitely, every set of consecutive measurements should be the same and have predictable parity 0 and be a detector. When a real experiment is run with errors, any detector with a parity different to that expected is called a detection event. See Fig. 5 for additional discussion and the definition of a detector graph.

The surface code admits an analogous example. If all data qubits are initialized to $|0\rangle$, then in the absence of errors all Z stabilizer measurements should report 0 and each of the first-round Z stabilizer measurements individually forms a detector. The first round of X stabilizer measurements will be random, so not detectors. When the second round of X and Z stabilizer measurements are performed, in the absence of errors the second round should match the first. The measurements from any two consecutive rounds can be paired up to give us a detector

on each measure qubit.

Consider a single X error between rounds of stabilizer measurement on the central qubit of a surface code (refer to Fig. 4). The next time the Z stabilizers are measured, the two Z stabilizers touching the central qubit will differ in sign from the previous round and two detection events will be generated. This X error will also flip the sign of Z_L. As such, not only can this X error be associated with an edge in the detector graph of the surface code, this edge can also be labeled as flipping the sign of Z_L. A Y error in the same place at the same time will anticommute with all four neighboring stabilizers, leading to four detection events. This can be associated with a hyperedge, so the surface code really has a detector hypergraph. This hyperedge will be labeled as flipping the sign of both Z_L and X_L. Given a detector hypergraph with a sufficiently sparse pattern of detection events, there are decoders that can find a high probability set of hyperedges that matches the detection events [23, 24]. This set tells us how many times the signs of the logical operators need to be flipped.

V. INITIALIZATION

Initialization to $|0_L\rangle$ means initialization of each data qubit to $|0\rangle$ followed by any number rounds of stabilizer measurement. There will be errors. Provided these errors are rare and sparse, they will lead to a sparse pattern of detection events that can be processed to select hyperedges in the surface code detector hypergraph that in turn tell us if the logical operators Z_L and X_L have been flipped.

It is worth pausing a moment to reflect what this means. Physical errors are not corrected. Just the flipping of logical operators is tracked. As errors occur, they can flip the signs of the stabilizers, and indeed after sufficient time the signs of all stabilizers will be random. For $|0_L\rangle$ initialization, the sign of X_L can be ignored since the surface code is not in an eigenstate of X_L. The state is defined by the sign of Z_L and the signs of the d^2-1 stabilizers. There are therefore 2^{d^2} states that can be used to represent $|0_L\rangle$.

If all of these states can represent $|0_L\rangle$, how does initialization fail? Consider X_L in Fig. 4. Suppose the top two data qubits along X_L in the $d=3$ code both suffer an X error. This flips the sign of Z_L, however only a single detection event associated with the bottom left Z stabilizer will be generated. The most likely explanation for this single detection event will be an X error on one of the bottom left two data qubits, both of which will be associated with a single detector graph edge to boundary. This edge will not suggest flipping the sign of Z_L, so after decoding we will believe the sign of Z_L has not flipped when in fact it has. This is a logical error.

If the circuits measuring the stabilizers have been designed correctly, it should take at least approximately $(d+1)/2$ independent physical errors to cause a logical error. Note from Fig. 3 that one measure qubit error can corrupt two data qubits. Such errors are called hook errors. It is very important that X stabilizer hook errors are perpendicular to X_L, otherwise only half as many errors are required to cause a logical error. As a technical aside, we saw in Fig. 6 that with sufficient variation in hyperedge probabilities, a decoder may choose more than the minimum number of hyperedges as the correction. When this choice is wrong, it is possible for fewer than $(d+1)/2$ errors to cause a logical error.

VI. MEASUREMENT

Measurement of the surface code in the X/Z basis means measuring each data qubit in that basis. Let's discuss measuring each data qubit in the Z basis. This is a destructive way to measure each Z stabilizer one final time. The X stabilizers are not measured. The parity of the individual Z measurements of each stabilizer's data qubits in the absence of errors will equal the same value as the measurement in the circuits of Fig. 3c. As with Z basis initialization, there will be one extra layer of detectors associated with the Z stabilizers. Each final detector will be a set of measurements consisting of the measurement from the last application of a Fig. 3c circuit and the measurements of the stabilizer's data qubits.

Measurements will of course contain errors and their raw values will be used to generate detection events that, as before, are used to select hyperedges that suggest flips of the logical operators. Fig. 5 shows a very simple example of constructing a detector hypergraph. In general you start with a quantum circuit annotated with detectors, then systematically consider every possible error on every gate, propagating the error until it generates detection events. These detection events become a hyperedge. If the hyperedge already exists, the probability of that hyperedge is increased. If the error also flips a logical operator, this information is recorded in the hyperedge. While perhaps not immediately obvious, every error that leads to the same hyperedge also leads to the same flipping of logical operators, provided $d \geq 3$ and the circuits of the code are well-formed. This is true since if two different error lead to the same detection events but flip different logical operators, then if both errors happen at the same time there will be no detection events but at least one logical operator will be flipped. This is another way of saying that those two errors form a logical operator, violating the assumption that the code distance is at least 3 and the circuits implementing the code are well-formed.

The detector hypergraph can be found using Stim [25]. Now would be a good time to work through Stim's getting started notebook. In Stim the logical operators are called observables, and you can only specify an observable with predictable value in the absence of errors. In the case of Z basis initialization this means logical measurement must also be in the Z basis to provide a predictable result

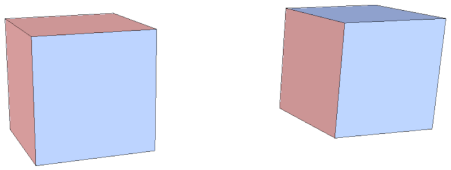

FIG. 7. Simple scalable representations of surface code memory experiments. Opposing faces of these cubes have identical colors. Red represents X, blue represents Z. Time runs vertically. A red/blue base means data qubit initialization in the X/Z basis. A red/blue top means data qubit measurement in the X/Z basis. Colored walls show the basis of the weight 2 stabilizers along that wall, and consequently the X/Z type of that boundary of the surface code. These cubes and their associated circuits can also be rotated 90° in space.

and only Z_L can be included as an observable.

The output of logical measurement is the parity of the raw physical measurements along the chosen logical operator, flipped an appropriate number of times as suggested by the set of hyperedges chosen by the decoder.

VII. MEMORY EXPERIMENTS

Consider once more Fig. 4. These circuits can be parameterized by k, where the central square of weight 4 stabilizers has dimensions $2k \times 2k$. The figure shows the circuits for $k = 1$ and $k = 2$, and defines the circuit for arbitrary k. For brevity, let's denote a member of this family of circuits by Mem_k. Let's define four variants of Mem_k: two variants that start with data qubit initialization in the Z/X basis, and two variants that end with data qubit measurement in the Z/X basis. Call these $InitX_k$, $InitZ_k$, $MeasX_k$, $MeasZ_k$. We can now define two memory experiments, $InitX_k$ followed by $2k-1$ rounds of Mem_k followed by $MeasX_k$, and $InitZ_k$ followed by $2k-1$ rounds of Mem_k followed by $MeasZ_k$. We can represent these very specific scalable memory experiments by simple colored cubes. See Fig. 7.

TODO: Discussion of state-of-the-art decoders not focusing on real-time.

TODO: A simulation of a memory experiment using the most promising decoder that might one day be real-time. Push down to very low error rates and very high code distances. Discuss low error rate limitations if found. Show equal weight advantage if possible.

TODO: Discuss using circuits with incorrect hook errors but alternating time direction to fix the code distance.

VIII. REAL-TIME DECODING

TODO: Discussion of state-of-the-art decoders focusing on real-time.

TODO: An extensive simulation of memory using this decoder.

TODO: Include a graph showing average latency as the decoder runs. Mention that only the latency of logical measurement really matters, since even in principle you don't do anything with anything else.

IX. IDENTITY WITH MOVEMENT

We have discussed memory experiments extensively, and the bulk of a memory experiment is a logical identity gate, but in this section we will upgrade our understanding of logical identity to include simple single-direction movements, see Fig. 8. This defines a precise 3D notation for both the underlying scalable circuits and the choice of physical and stabilizer measurements and initializations to move logical operators.

Each cube in Fig. 8f should initially be thought of as an $InitZ_k$–$(2k-1)\times Mem_k$–$MeasZ_k$ memory experiment. The long pipes modify the walls of these experiments. The first vertical pipe should be interpreted as an instruction to replace $MeasZ_k$ in the first cube and $InitZ_k$ in the second cube with Mem_k layers. The horizontal pipe replaces the boundary walls of the two cubes it touches with connecting stabilizer measurements, along with appropriate data qubit initialization and measurement.

Fig. 8 contains more detail than we would like to continue discussing moving forward. Fig. 9 focuses on the mapping of X_L and Z_L at input to X'_L and Z'_L at output and defines correlation surfaces and byproduct operators. Fortunately, software exists [26] to find correlation surfaces and calculate the logical error rate of computations specified as geometric structures.

X. CNOT

The definition of CNOT given in Section I is a called a set of stabilizer flows. We will multiply the third into the first and the second into the fourth to obtain an equivalent but more convenient set.

$$X_aX_b \xrightarrow{CNOT_{ab}} X_aI_b$$
$$Z_aI_b \xrightarrow{CNOT_{ab}} Z_aI_b$$
$$I_aX_b \xrightarrow{CNOT_{ab}} I_aX_b$$
$$Z_aZ_b \xrightarrow{CNOT_{ab}} I_aZ_b$$

A structure with correlation surfaces matching these stabilizer flows will implement CNOT up to byproduct operators [27]. Fig. 10 shows one such structure using lattice surgery [28, 29]. Let $(X_aI_b)^{\lambda_{Z_a}}(Z_aI_b)^{\lambda_{X_a}}(I_aX_b)^{\lambda_{Z_b}}(I_aZ_b)^{\lambda_{X_b}}$ be the input byproduct operators. The notation is intended to make it clear, for example, that the corrected parity of the Z

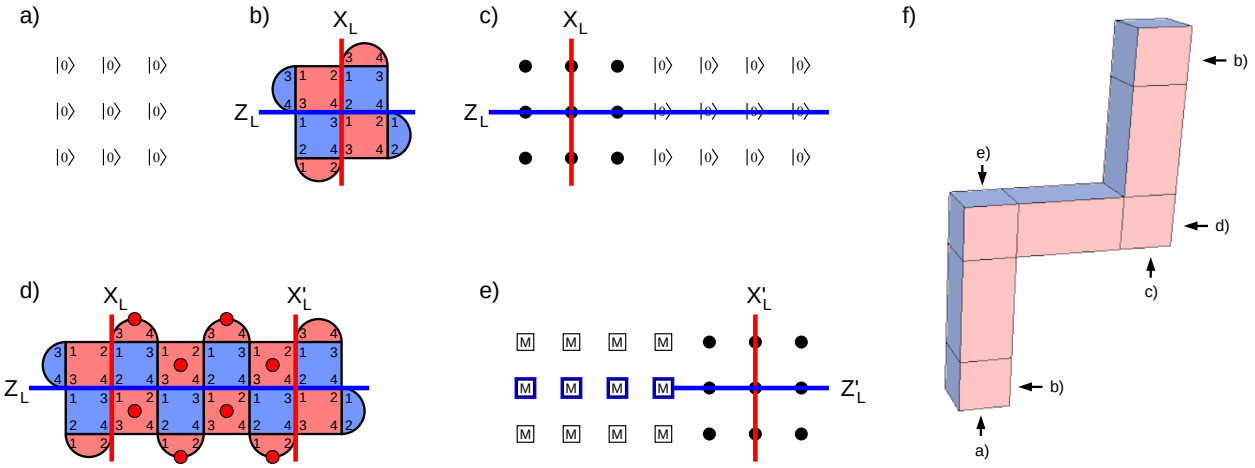

FIG. 8. a) All data qubits initialized to $|0\rangle$. b) $2k+1$ rounds of stabilizer measurement. c) Beginning to extend the logical qubit with more data qubits initialized to $|0\rangle$. Black dots represent data qubits doing nothing. Z_L can be extended without sign change across these $|0\rangle$ values. Any errors and consequent flips of Z_L can temporarily be ignored as there are no detection events yet. d) $2k+1$ rounds of stabilizer measurement during which stabilizers indicated with red dots are used to move X_L. The parity of any chosen round of these measurements sets an initial sign relationship between X_L and X'_L. Our convention is to choose the earliest round. e) Z basis measurement of data qubits. The parity of the blue highlighted raw values sets an initial sign relationship between Z_L and Z'_L. Both X'_L and Z'_L can be flipped by the choice of hyperedges matching any detection events. f) Simple and precise visual representation of the circuits described in parts a-e and accompanying text. Opposing boundaries have the same color.

correlation surface attached to input a (λ_{Za}) controls the presence of a byproduct X on input a (X_aI_b). Let λ'_{Xa}, λ'_{Za}, λ'_{Xb}, λ'_{Zb} be the corrected parities of the correlation surfaces of Figs. 10b-e, respectively. Focusing on the output byproduct operator Z_aI_b, this will depend on λ'_{Xa} and the two connected inputs λ_{Xa} and λ_{Xb}. The output byproduct operators will be

$$(X_aI_b)^{\lambda'_{Za}+\lambda_{Za}}$$
$$(Z_aI_b)^{\lambda'_{Xa}+\lambda_{Xa}+\lambda_{Xb}}$$
$$(I_aX_b)^{\lambda'_{Zb}+\lambda_{Za}+\lambda_{Zb}}$$
$$(I_aZ_b)^{\lambda'_{Xb}+\lambda_{Xb}}.$$

XI. JUNCTIONS AND CORNERS

Given Fig. 8 and Fig. 10, it may seem like we have already covered junctions and corners, since these figures do indeed contain both and correspond to very specific quantum circuits with well-defined detectors and correlation surfaces. However, these figures only covered the simple case, when corners and junctions occur in time. A junction or corner in space requires considerably more care due to hook errors and maintaining code alignment.

Code alignment refers to where the weight 2 stabilizers are located. In Fig. 4, as we go around the boundary clockwise starting on the top left corner, every second location has a weight 2 stabilizer. We shall call this even alignment, or an even surface code. A surface code can have odd alignment, with weight 2 stabilizers starting on the first position, however to facilitate consistent and simple connection of 3D objects, we shall forbid the use of odd surface codes. Fig. 11 shows a spatial 4-way junction in detail. To obtain T- and L-junctions, we start with a 4-way junction and modify the circuit as described in Fig. 12. Spatial junctions and corners with X boundaries can be obtained simply by interchanging the red and blue colors in these figures.

XII. ZX-CALCULUS

Now that we know how to construct specific scalable circuits for junctions and corners of any orientation, it is time to stop doing so. Consider Fig. 13. This provides a graphical shorthand focusing on how correlation surfaces are affected by different junctions [30, 31]. Given a bounded region, two graphs inside that region are equivalent if they have the same number of edges crossing the boundary and the correlation surfaces supported on these edges are identical. A few useful graph identities can be found in Fig. 14.

Let's work through an example. Consider the three CNOT circuit of Fig. 15a. A direct implementation of this using Fig. 10 is shown in Fig. 15e. If we convert

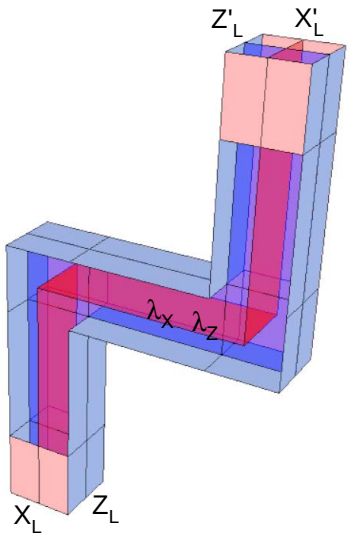

FIG. 9. A version of Fig. 8 with details suppressed and simplified internals exposed. The translucent surfaces are called correlation surfaces. The red (X) and blue (Z) correlation surfaces must terminate on boundaries of the same color. A correlation surface represents all physical initializations, measurements, and stabilizer measurements required to map logical operators from input to output. λ_X and λ_Z represent the corrected parity of everything in the respective correlation surface. $X'_L = (-1)^{\lambda_X} X_L$. $Z'_L = (-1)^{\lambda_Z} Z_L$. Equivalently, we can say the output differs from the ideal output by a byproduct operator $Z^{\lambda_X} X^{\lambda_Z}$.

this to a ZX graph, then use the spider rule twice and rearrange, we end up with Fig. 15d, which can be implemented compactly as Fig. 15f. This structure can be rotated arbitrarily in space and time, and will still implement the same computation as the original three CNOT circuit, up to byproduct operators that can be calculated from the correlation surfaces provided in Fig. 16.

TODO: Make it possible to simulate this using the TQEC tool. Explain how to do so. Recommend reader do so.

XIII. HADAMARD AND CZ

Logical Hadamard can be implemented in space or time with effectively zero cost, see Fig. 17. Hadamard can in turn be used to implement logical CZ with very low overhead, see Fig. 18. Note that

$$\begin{aligned} X_a Z_b &\xrightarrow{CZ_{ab}} X_a I_b \\ Z_a I_b &\xrightarrow{CZ_{ab}} Z_a I_b \\ Z_a X_b &\xrightarrow{CZ_{ab}} I_a X_b \\ I_a Z_b &\xrightarrow{CZ_{ab}} I_a Z_b. \end{aligned}$$

Readers should verify Figs. 18b-c support these correlation surfaces, and construct a similar low overhead CNOT occupying the same space-time volume.

XIV. PATCH ROTATION

When logical qubits are moved diagonally, they can reach their destination 90° rotated, or not, as desired. The determining factor is the number of spatial corners included along the path. See Fig. 19. Frequently, however, 90° rotations are desired without significant movement. This can be achieved using a $1\times2\times2$ structure with specific boundaries as shown in Fig. 20, which can be rotated arbitrarily in space and time. There are a number of details that need to be discussed. For brevity of discussion, we will focus on this one orientation of patch rotation, and the reader is encouraged to work out the details for at least one other orientation of this structure.

Implementing Fig. 20 requires maintaining surface code alignment and avoiding dangerous hook errors. Hook errors are particularly important to avoid in small surface codes. Let's start with defining four plaquettes (Fig. 21). These plaquettes can be tiled arbitrarily in a checkerboard, giving one total control over hook errors. We can use these plaquettes to implement patch rotation with full code distance, see Fig. 22.

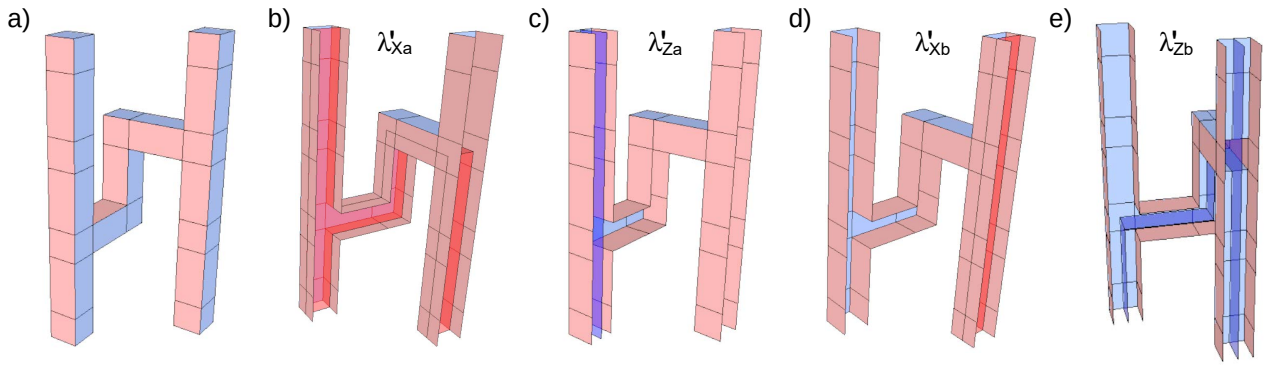

FIG. 10. a) Logical CNOT with left column control and right column target. b-e) Cutaway models showing each correlation surface implementing the CNOT stabilizer flows.

[1] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Phys. Rev. A **86**, 032324 (2012), arXiv:1208.0928.

[2] Y. I. Manin, *Computable and Uncomputable* (Sovetskoe Radio, Moscow, 1980).

[3] P. Benioff, J. Stat. Phys. **22**, 563 (1980).

[4] R. P. Feynman, Int. J. Theor. Phys. **21**, 467 (1982).

[5] D. Deutsch, Proc. R. Soc. Lond. A **400**, 97 (1985).

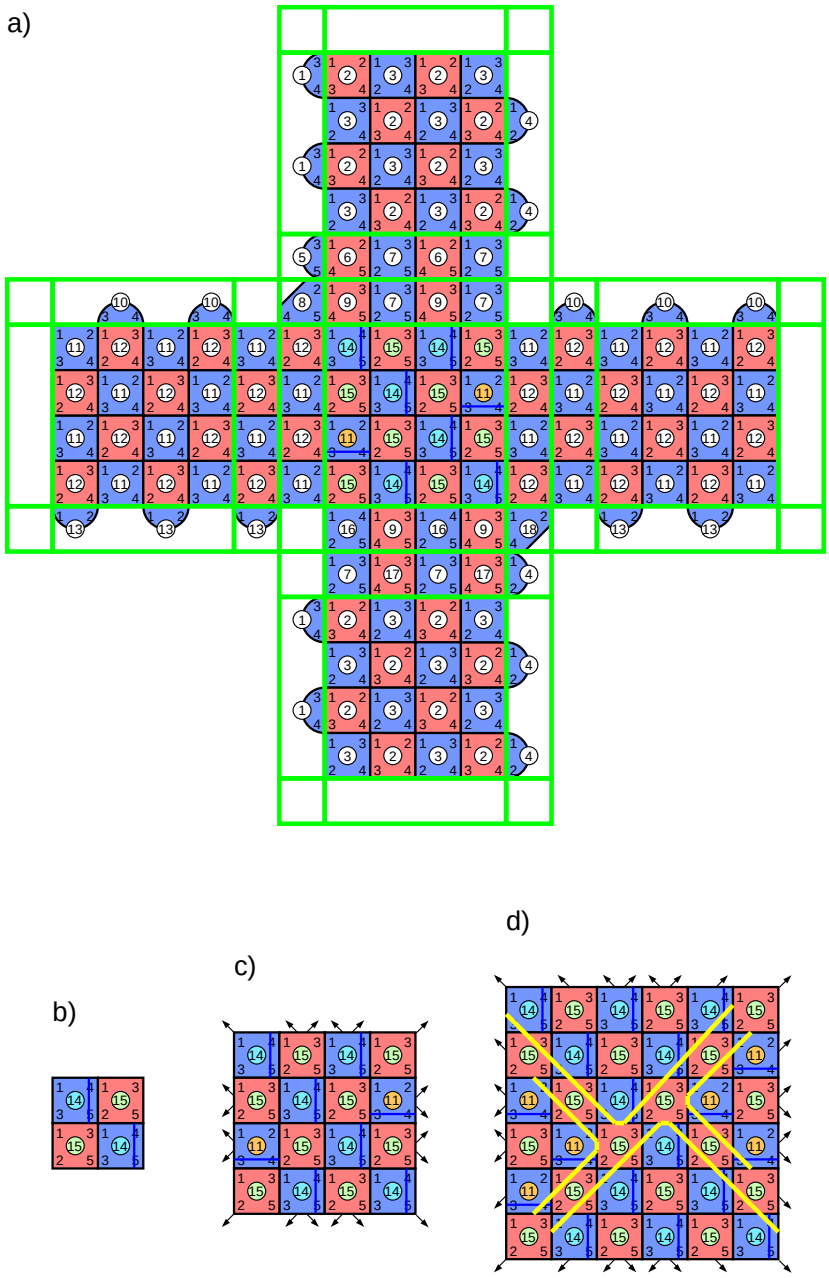

FIG. 11. a) A stabilizer measurement layer of a $k = 2$ (distance $d = 5$) even surface code 4-way spatial junction showing all details. Green lines show 1×1, 2k×1, and 2k×2k plaquette tilings that respectively do not scale, scale in 1D, or scale in 2D with k. Plaquettes with the same number contain the same circuit. The double row of X and Z plaquettes above and below the center individually produce random results since each anticommutes with at least one other plaquette, but vertical pairs of plaquettes commute with neighboring pairs of plaquettes and therefore these pairs can be compared with pairs later and earlier to form detection events. These double rows effectively contain stretched stabilizers used to keep the surface code even everywhere. b-d) Sequence of junction centers showing how to systematically increment k and grow the circuit and associate code distance $d = 2k + 1$. Plaquettes 11 and 14 can be interchanged to rotate the direction of hook errors. These two types of circuits are arranged in four growing triangles pointing towards the center. This arrangement ensures that it takes at least the code distance number of errors to form an undetectable error chain stretching from any corner of the center to any other, and by extension from any part of the boundary to any other, since this added boundary only touches this center square on the corners and the bulk of the arms of the junction have hook errors aligned perpendicular to the logical operators. Note that both the center and double layer circuits require 5 times steps to schedule the two-qubit gates so that hook errors do not degrade the code distance and neighboring plaquettes are strictly time ordered, the latter feature being required for successful simultaneous measurement.

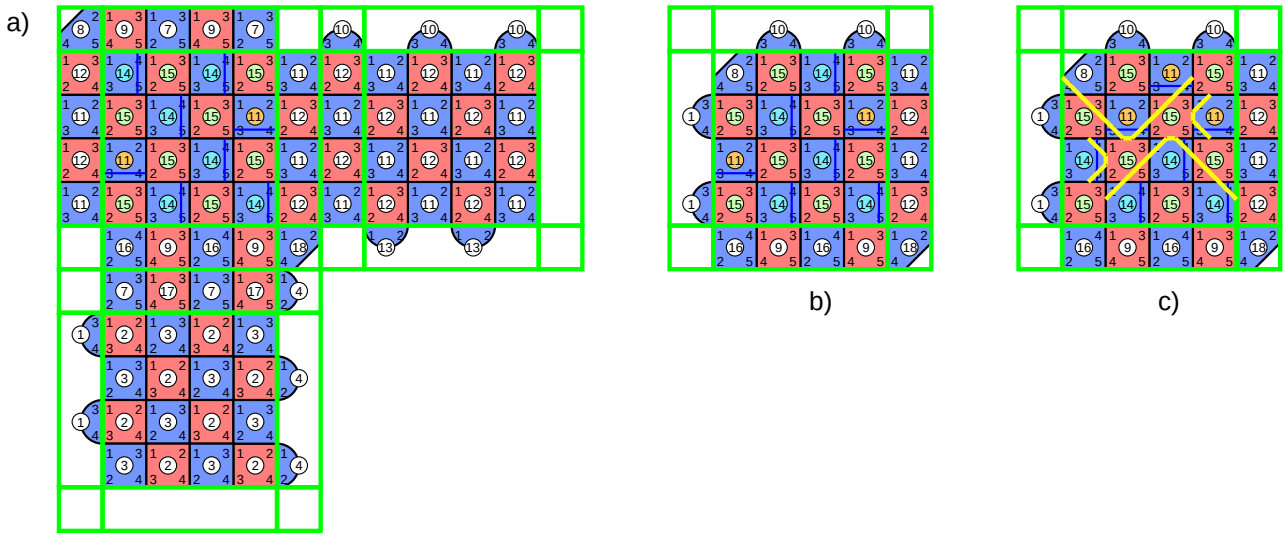

FIG. 12. a) Fig. 11 with two arms removed. b) The corner with the boundary trimmed. c) The triangular regions of plaquettes adjacent to boundaries have been modified so that their dangerous hook errors are parallel to the boundary. In this specific case, in the top triangle plaquettes 14 have become 11 and in the left triangle plaquettes 11 have become 14. These errors on the measure qubits of these Z plaquettes can result in Z errors on both qubits indicated by the blue bar. Since we must make sure that there is no easy way to form chains of Z errors from boundary to boundary, these pairs of correlated errors must be arranged with care.

[6] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, Phys. Rev. A **52**, 3457 (1995), quant-ph/9503016.

[7] E. Muñoz-Coreas and H. Thapliyal, in *Wiley Encyclopedia of Electrical and Electronics Engineering*, edited by J. G. Webster (Wiley, 2024) arXiv:2208.11725.

[8] W. Pauli, Zeitschrift für Physik **43**, 601 (1927).

[9] D. Gottesman, Phys. Rev. A **57**, 127 (1998), quant-ph/9702029.

[10] `https://en.wikipedia.org/wiki/Tensor_product`.

[11] D. Gottesman, *Stabilizer Codes and Quantum Error Correction*, Ph.D. thesis, Caltech (1997), quant-ph/9705052.

[12] K. C. Miao, M. McEwen, J. Atalaya, D. Kafri, L. P. Pryadko, A. Bengtsson, A. Opremcak, K. J. Satzinger, Z. Chen, P. V. Klimov, C. Quintana, R. Acharya, K. Anderson, M. Ansmann, *et al.*, Nat. Phys. **19**, 1780– (2023), arXiv:2211.04728.

[13] M. McEwen, K. C. Miao, J. Atalaya, A. Bilmes, A. Crook, J. Bovaird, J. M. Kreikebaum, N. Zobrist, E. Jeffrey, B. Ying, A. Bengtsson, H.-S. Chang, A. Dunsworth, J. Kelly, Y. Zhang, E. Forati, R. Acharya, J. Iveland, W. Liu, S. Kim, B. Burkett, A. Megrant, Y. Chen, C. Neill, D. Sank, M. Devoret, and A. Opremcak, arXiv:2402.15644 (2024).

[14] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica, G. Salton, S. Wang, and F. G. S. L. Brandão, arXiv:2310.03011 (2023).

[15] D. Gottesman, arXiv:1310.2984 (2013).

[16] Q. Xu, J. P. B. Ataides, C. A. Pattison, N. Raveendran, D. Bluvstein, J. Wurtz, B. Vasic, M. D. Lukin, L. Jiang, and H. Zhou, Nat. Phys. **20**, 1084– (2024), arXiv:2308.08648.

[17] S. B. Bravyi and A. Y. Kitaev, quant-ph/9811052 (1998).

[18] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, J. Math. Phys. **43**, 4452 (2002), arXiv:quant-ph/0110143.

[19] R. Raussendorf and J. Harrington, Phys. Rev. Lett. **98**, 190504 (2007), quant-ph/0610082.

[20] R. Raussendorf, J. Harrington, and K. Goyal, New J. Phys. **9**, 199 (2007), quant-ph/0703143.

[21] R. Acharya *et al.*, arXiv:2408.13687 (2024).

[22] C. Gidney, M. Newman, A. Fowler, and M. Broughton, Quantum **5**, 605 (2021), arXiv:2108.10457.

[23] C. Jones, arXiv:2408.12135 (2024).

[24] K.-Y. Kuo and C.-Y. Lai, arXiv:2409.18689 (2024).

[25] C. Gidney, Quantum **5**, 497 (2021), arXiv:2103.02202.

[26] `github.com/QCHackers/tqec/`.

[27] R. Raussendorf, D. E. Browne, and H.-J. Briegel, Phys. Rev. A **68**, 022312 (2003), quant-ph/0301052.

[28] D. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, New J. Phys. **14**, 123011 (2012), arXiv:1111.4022.

[29] A. G. Fowler and C. Gidney, arXiv:1808.06709.

[30] B. Coecke and A. Kissinger, *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning* (Cambridge University Press, Cambridge, 2017).

[31] J. van de Wetering, arXiv:2012.13966 (2020).

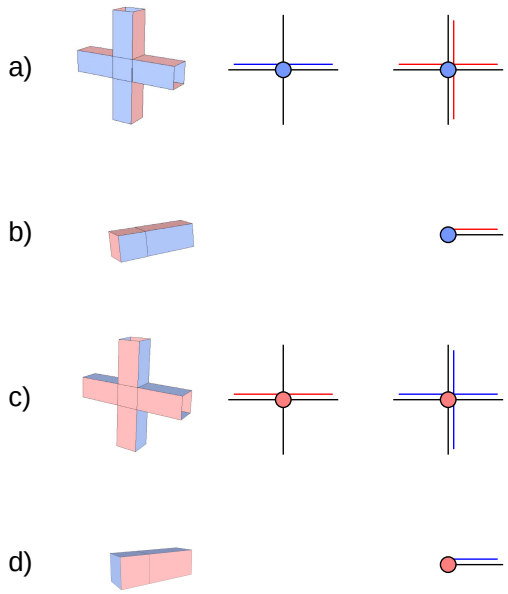

FIG. 13. a) Blue (Z) nodes can have a blue (Z) correlation surface on exactly two edges, but red (X) correlation surfaces must be on all edges. b) A blue node with a single edge doesn't support a blue correlation surface. c) Red (X) nodes can have a red (X) correlation surface on exactly two edges, but blue (Z) correlation surfaces must be on all edges. d) A red node with a single edge doesn't support a red correlation surface. Single edge nodes can be thought of as measurement or initialization in the basis of the supported correlation surface. Many previous works have used black instead of blue nodes and white instead of red nodes, or a green and red rather than a blue and red convention, but we will maintain our XYZ=RGB mnemonic for consistency.

Appendix A: Projecting general errors to Pauli products

Consider Fig. 2 with erroneous input $E|\Psi\rangle$, $E = T\otimes T$, $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, and $A = X \otimes X$. To be explicit,

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4}. \end{bmatrix} \tag{A1}$$

So $T|0\rangle = |0\rangle$ and $T|1\rangle = e^{i\pi/4}|1\rangle$. Let's work this through step-by-step.

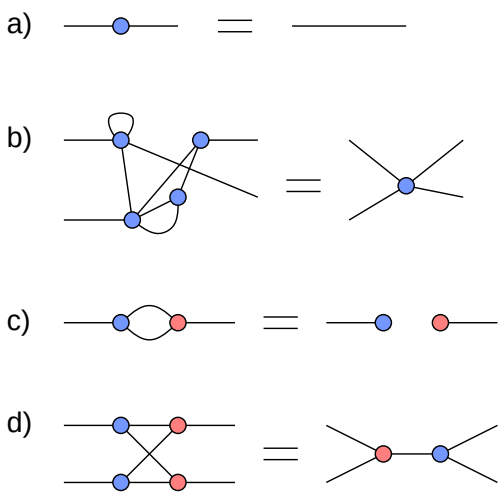

FIG. 14. ZX calculus graph identities. All identities are valid with blue and red interchanged. a) Identity rule: nodes on a simple path can be added or removed as they do not modify correlation surfaces. b) Spider rule: any connected graph of same color nodes can be reduced to a single node. c) Hopf law: pairs of edges connecting different color nodes can be removed. d) Bialgebra law.

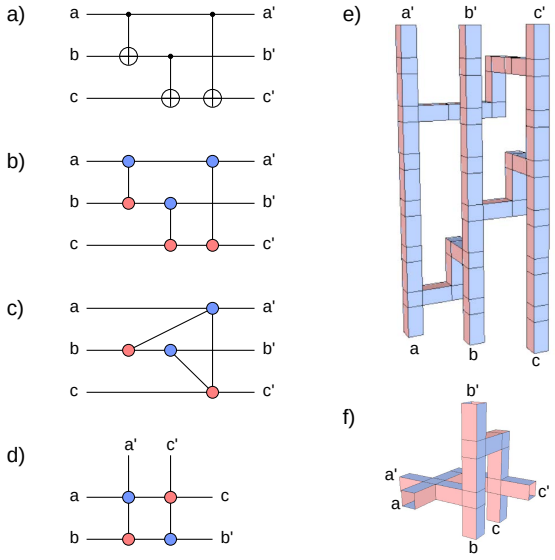

FIG. 15. a) Three CNOT circuit. b) Same circuit expressed as a ZX graph. c) Graph after two applications of the spider rule. d) Rearrangement of the graph. e) Direct implementation of the three CNOT circuit using Fig. 10. f) Direct implementation of the final ZX graph.

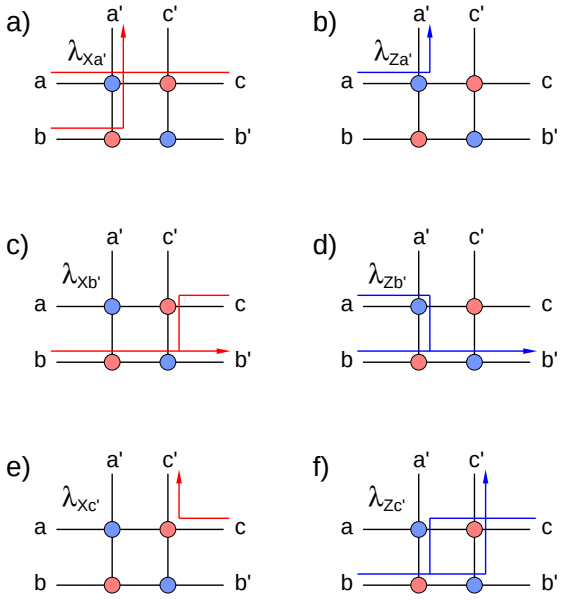

FIG. 16. Correlation surfaces of Fig. 15d relating each output logical operator to a combination of input logical operators for easy byproduct operator calculation.

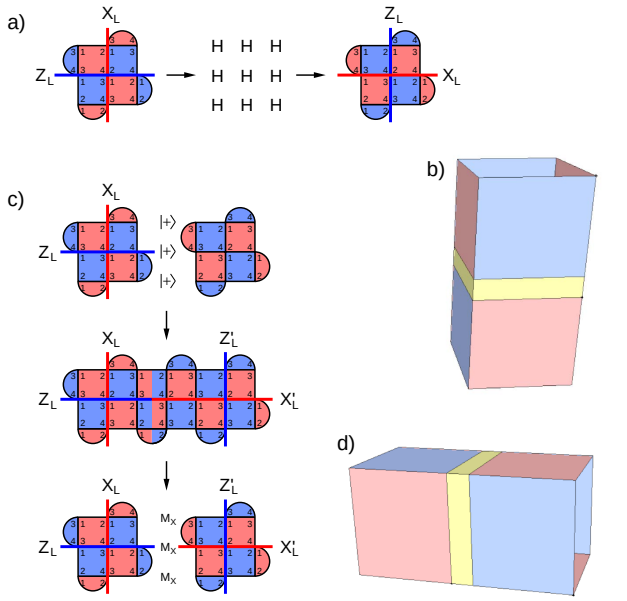

FIG. 17. a) Logical Hadamard performed in time. Physical Hadamard is applied to each data qubit after a round of surface code stabilizer measurement and before a round of 90° rotated stabilizer measurement. b) A 3D connector representing logical Hadamard in time. The physical Hadamard gates can be incorporated into either of the two rounds of stabilizer measurements this connector represents. These two rounds replace the final and initial rounds in the cubes they connect. As such, logical Hadamard has effectively zero cost. C) Logical Hadamard performed in space. The focus is on the center two columns of stabilizers. The two-color stabilizers are a mix of X and Z and logical operators that move through them change type. These two columns replace the stabilizers that were on the walls of the cubes they connect, and again there is zero cost. d) 3D representation of logical Hadamard performed in space.

$$
\begin{aligned}
&|0\rangle E |\Psi\rangle \\
&= |0\rangle T \otimes T \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle) \\
&= |0\rangle \frac{1}{\sqrt{2}} (|00\rangle + i|11\rangle) \\
&\xrightarrow{H} |+\rangle \frac{1}{\sqrt{2}} (|00\rangle + i|11\rangle) \\
&\xrightarrow{ctrl-X\otimes X} \frac{1}{\sqrt{2}} \left(|0\rangle \frac{1}{\sqrt{2}} (|00\rangle + i|11\rangle) + \right. \\
&\quad \left. |1\rangle \frac{1}{\sqrt{2}} (|11\rangle + i|00\rangle) \right) \\
&\xrightarrow{H} \frac{1}{\sqrt{2}} \left(|+\rangle \frac{1}{\sqrt{2}} (|00\rangle + i|11\rangle) + \right. \\
&\quad \left. |-\rangle \frac{1}{\sqrt{2}} (|11\rangle + i|00\rangle) \right)
\end{aligned}
$$

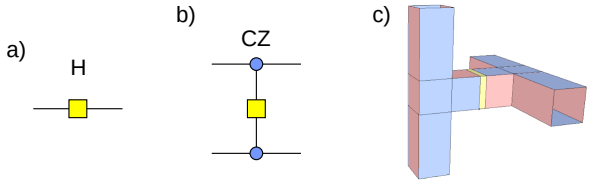

FIG. 18. a) ZX graph for Hadamard. b) ZX graph for CZ. c) Low-overhead CZ realization. The inputs can be any two ports on opposite sides of the Hadamard.

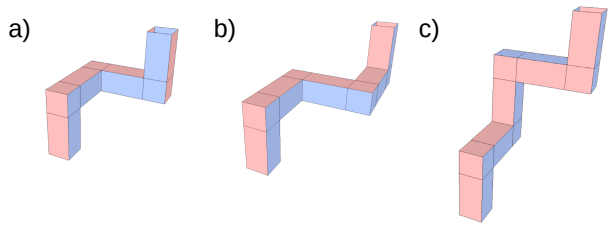

FIG. 19. a) A diagonal move with one spatial corner resulting in the logical qubit being rotated 90°. b) A diagonal move with two spatial corners and hence no rotation of the logical qubit. c) A diagonal move with no spatial corners and hence no rotation of the logical qubit.

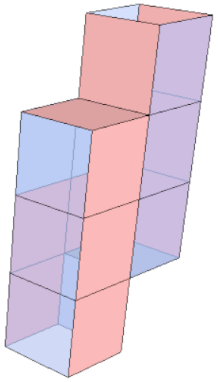

FIG. 20. Boundary structure of a 1×2×2 box that rotates a logical qubit by 90°. Z boundaries have been made translucent to enable a single viewpoint to show all boundary types.

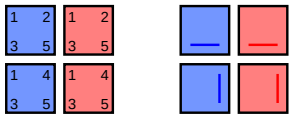

FIG. 21. Four unique plaquettes that can be tiled arbitrarily in a checkerboard, giving one total control over hook errors. Note that the final two qubits touched are aligned horizontally on the first row and vertically on the second row. An error on the measure qubit during stabilizer measurement can propagate to both of these final two qubits. Plaquettes on the left show the timing of the two-qubit gates, those on the right represent these in a visually simpler manner.

$$
\begin{aligned}
&= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}|0\rangle\frac{1}{\sqrt{2}}\left(|00\rangle + i|11\rangle + |11\rangle + i|00\rangle\right) + \right.\\
&\quad \left.\frac{1}{\sqrt{2}}|1\rangle\frac{1}{\sqrt{2}}\left(|00\rangle + i|11\rangle - |11\rangle - i|00\rangle\right)\right)\\
&= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}|0\rangle\frac{1}{\sqrt{2}}\left((1+i)|00\rangle + (1+i)|11\rangle\right) + \right.\\
&\quad \left.\frac{1}{\sqrt{2}}|1\rangle\frac{1}{\sqrt{2}}\left((1-i)|00\rangle + (-1+i)|11\rangle\right)\right)\\
&= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}|0\rangle\left(e^{i\pi/4}|00\rangle + e^{i\pi/4}|11\rangle\right) + \right.\\
&\quad \left.\frac{1}{\sqrt{2}}|1\rangle\left(e^{-i\pi/4}|00\rangle + e^{i3\pi/4}|11\rangle\right)\right)
\end{aligned}
$$

If the measurement $M = 0$, the resultant state will be

$$\frac{1}{\sqrt{2}}\left(e^{i\pi/4}|00\rangle + e^{i\pi/4}|11\rangle\right) = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right).$$

Equality follows from ignoring the irrelevant global phase $e^{i\pi/4}$. Note that this is precisely the error-free state $|\Psi\rangle$. If $M = 1$ the resultant state will be

$$\frac{1}{\sqrt{2}}|1\rangle\left(e^{-i\pi/4}|00\rangle + e^{i3\pi/4}|11\rangle\right) = \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right).$$

Equality follows by multiplying by an irrelevant global phase of $e^{i\pi/4}$. In this case, the output state $|\Psi'\rangle = ZI|\Psi\rangle = IZ|\Psi\rangle$. The error has been projected to a simple Pauli error. Note that the symmetry of the state means that it doesn't matter whether we consider the Z error to have occurred on the first or second qubit.

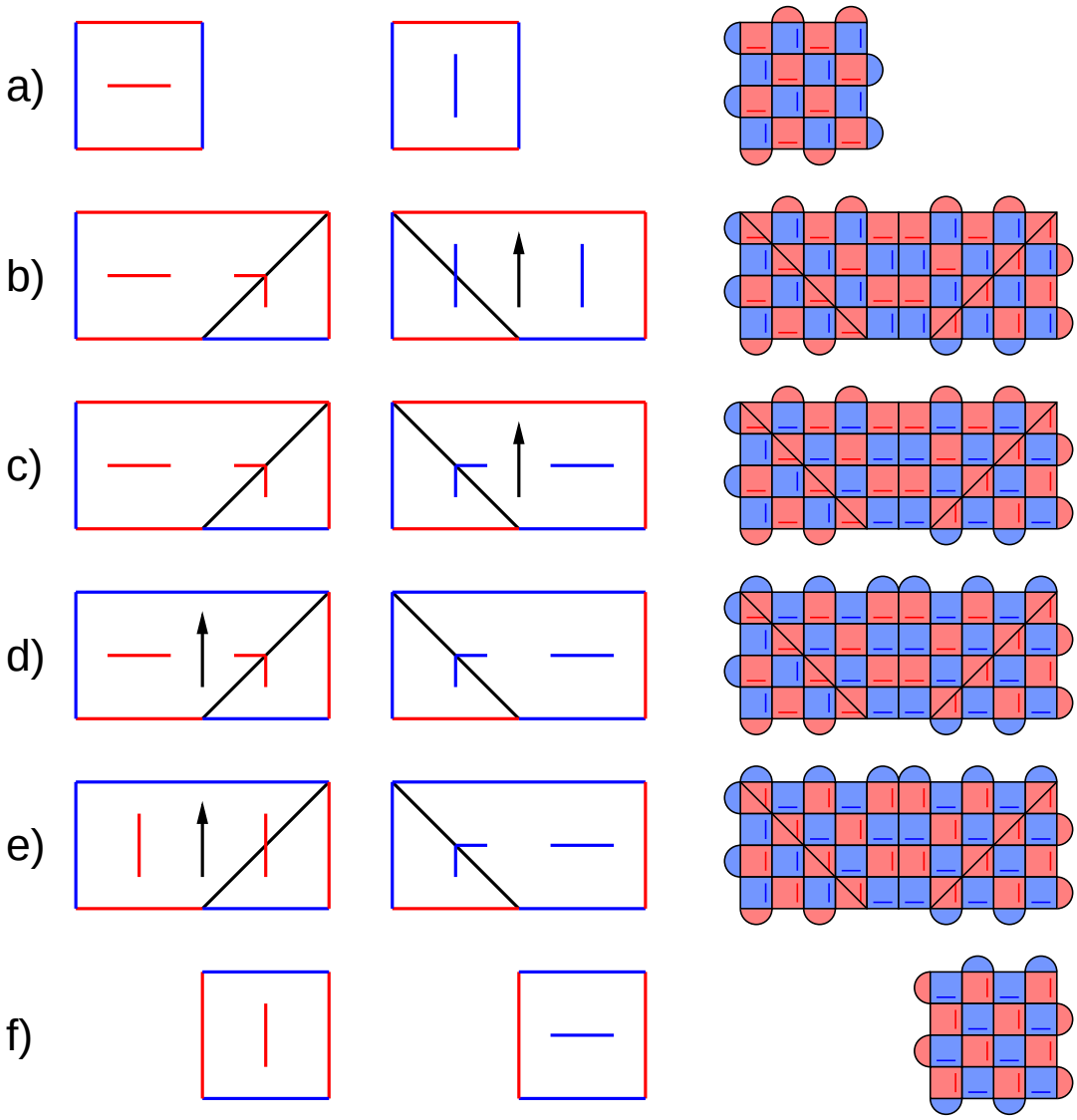

FIG. 22. a) Initial configuration of the surface code logical qubit. The left figure shows the direction of X hook errors, the middle figure the direction of Z hook errors. The right figure gives an explicit example at $d = 5$. Note that hook errors are parallel to pairs of boundaries of the same type to avoid low weight paths of errors connecting these boundaries. b) Expanding the logical qubit to the right. Note the need to adjust the orientation of X hook errors as the bottom red boundary must be kept well separated from the right red boundary. c) Over d rounds, starting on the bottom row of plaquettes, the orientation of Z hook errors is rotated in preparation for the sudden change in boundary type in part d. This orientation can't be used immediately in part b as it would lead to short error chains connecting boundaries in part a. d) Boundary types changed, no hook error orientations changed. e) Over d rounds, starting on the bottom row of plaquettes, the orientation of X hook errors is rotated in preparation for the sudden change in boundary type in part f. This orientation can't be used immediately in part d as it would lead to short error chains connecting boundaries in part c.