# Poking Holes and Cutting Corners to Achieve Clifford Gates with the Surface Code

Benjamin J. Brown,[1,*] Katharina Laubscher,[2] Markus S. Kesselring,[2,3] and James R. Wootton[2]

[1]*Niels Bohr International Academy, Niels Bohr Institute, Blegdamsvej 17, 2100 Copenhagen, Denmark*
[2]*Department of Physics, University of Basel, Klingelbergstrasse 82, CH-4056 Basel, Switzerland*
[3]*Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany*

The surface code is currently the leading proposal to achieve fault-tolerant quantum computation. Among its strengths are the plethora of known ways in which fault-tolerant Clifford operations can be performed, namely, by deforming the topology of the surface, by the fusion and splitting of codes, and even by braiding engineered Majorana modes using twist defects. Here, we present a unified framework to describe these methods, which can be used to better compare different schemes and to facilitate the design of hybrid schemes. Our unification includes the identification of twist defects with the corners of the planar code. This identification enables us to perform single-qubit Clifford gates by exchanging the corners of the planar code via code deformation. We analyze ways in which different schemes can be combined and propose a new logical encoding. We also show how all of the Clifford gates can be implemented with the planar code, without loss of distance, using code deformations, thus offering an attractive alternative to ancilla-mediated schemes to complete the Clifford group with lattice surgery.

Subject Areas: Quantum Information

## I. INTRODUCTION

The components that make up quantum technologies are inherently sensitive to noise. This is a problem that, if unresolved, will inhibit the scalability of quantum information processing tasks. To overcome this issue, quantum error-correcting codes have been developed [1–5], where logical quantum states are redundantly encoded in robust subspaces of the Hilbert space of many physical qubits. As such, there is a considerable number of physical qubits of a scalable quantum computer that are dedicated to the role of error correction.

Given the prohibitive cost of quantum resources, it is important to discover fault-tolerant schemes for universal quantum computation that use as few physical qubits as possible. Among other factors, the resource cost of fault-tolerant quantum computation depends on the choice of quantum error-correcting codes into which we choose to encode quantum information, as well as the different schemes we use to implement a universal set of computational gates. Indeed, there has been considerable effort dedicated to minimizing the resource cost of quantum computation.

A leading approach to realizing fault-tolerant quantum computation is broadly known as topological quantum computation [1,6–8]. With this approach, we protect quantum information by encoding it into nonlocal degrees of freedom, using objects including non-Abelian anyons [1,9] and punctures [10–19], or by use of extrinsic defects [20–25], otherwise known as twists. With these schemes, encoded information is manipulated by braiding these objects to realize fault-tolerant universal quantum computation. In addition to schemes where protected quantum information undergoes unitary rotations by braiding, other promising schemes are known where quantum information is encoded over noncontractable cycles of a lattice of physical qubits, which are embedded on a manifold with nontrivial topology [1,2]. Fault-tolerant entangling gates are then achieved either transversally [26–28] or by lattice surgery [29,30].

It is the goal of this manuscript to unify some of these schemes by consideration of a specific lattice model. Here we unify two low-overhead approaches to encoding qubits using the surface code [1,2,31], namely, lattice surgery [29,30], together with defect encoding schemes [20,24]. This unification is made using a correspondence between the corners of the planar code [2] and twist defects [20].

While it is well known in generality that non-Abelian defects exist on system boundaries in between distinct phases in the topological condensed-matter literature [23,32–36], here we find it instructive to consider boundaries of a very specific lattice system. In particular, we use the correspondence between twist defects and Majorana fermions [20,25,37,38] to realize logical gates by braiding twist defects. Specifically, we show that we can achieve a fault-tolerant realization of the full Clifford group in a two-dimensional system using lattice surgery methods, and by braiding twist defects via code deformations.

---

*benjamin.brown@nbi.ku.dk

Furthermore, we also consider interactions between different topological schemes for encoding quantum information. In particular, we consider how punctures interact with twist defects and show that we can use holes to perform fault-tolerant measurement-only topological quantum computation [24,39,40] with twist defects, by braiding twists and holes. We also show that we can encode logical qubits in hybridized schemes that use both holes and twist defects to encode logical qubits. Such encodings are of interest as they may enable new schemes that realize fault-tolerant quantum computation with lower overhead demands. See, for instance, related recent work [41] following this goal, where logical qubits are encoded on a surface code with three corners and a single central twist.

The paper is organized as follows: In Sec. II, we give a brief overview of some of the different approaches that have been followed to achieve low-overhead fault-tolerant quantum computation. Then, in Sec. III, we introduce notation and review the stabilizer formalism, anyon models, the Clifford group, and different methods of encoding qubits using the surface code. We also develop a diagrammatic language that we use to build correspondences between different schemes for encoding qubits. In Sec. IV, we review code deformation and show that we can braid the corners of the planar code by code deformation to perform Clifford gates. In Sec. V, we consider interactions between twists and holes, and show that we can use holes to perform fault-tolerant parity measurements between qubits encoded using twist defects in a measurement-only topological quantum-computational scheme. In Sec. VI, we illustrate a connection between lattice surgery and measurement-only topological quantum computation. Finally, in Sec. VII, we introduce a new hybrid scheme of encoding qubits and discuss its advantages and drawbacks before giving some concluding remarks in Sec. VIII. We point out that further details and alternative explanations of some aspects of this manuscript can be found in the work on which it is based (see Ref. [42]).

## II. FAULT-TOLERANT TOPOLOGICAL QUANTUM COMPUTATION SCHEMES

This work seeks to develop and unify tools for surface-code quantum computation. Before we present our results, we first comment on other promising topological quantum-computational schemes. Ultimately, the resource cost of a computational scheme will depend on how the logical error rate of a code scales as a function of the number of physical qubits used, together with time and space requirements that are needed to execute a universal gate set with the chosen system. Certainly, it is important to consider all of these factors when trying to determine the number of physical qubits a particular architecture will need.

Given a system that can perform Clifford gates and prepare noisy copies of magic states, universal quantum computation can be achieved via magic-state distillation [43]. A recent review of developments in magic-state distillation protocols can be found in Ref. [44]. As such, we first restrict our attention to achieving Clifford gates in two-dimensional architectures.

Previously, using the surface code where qubits are encoded using punctures, the full Clifford group is completed using ancillas that are prepared in the $Y$ state [16], i.e., an eigenstate of the Pauli-$Y$ matrix. Logical qubits are prepared in the $Y$ state via a probabilistic and noisy process [45]. Once prepared, the distilled states can be used to perform an arbitrary number of phase gates [46,47]. Nevertheless, the requirement for nearby ancilla qubits prepared in the $Y$ states, together with the initial overhead cost of preparing these $Y$ states, will contribute to the resource cost of quantum computation compared with schemes where phase gates can be achieved natively. Alternatively, by the addition of lattice dislocations [24], the Clifford group can be achieved by making parity measurements between logical qubits. Indeed, given arbitrary two-qubit Pauli parity measurements, arbitrary two-qubit Clifford gates can be achieved [4]. The ability to perform arbitrary two-qubit gates circumvents the need for single-qubit Clifford gates to generate the Clifford group. Notably, the dislocation code scheme involves introducing a small number of weight-five stabilizer measurements to the surface code.

The two-dimensional color code also achieves the full Clifford group logical gates via transversal operations [26]. While transversal operations are very appealing compared with the code-deformation schemes that seem to be required for quantum computation with the surface code, realizing the color code comes at the expense of increased weight stabilizer measurements. Specifically, the surface code requires weight-four stabilizer measurements compared with weight-six measurements that are required of the color code.

It is also worth mentioning the gauge color code [28], which, notably, performs a universal transversal gate set via gauge fixing [48,49] and may thus offer a reduction in overhead compared with magic-state distillation-based schemes of computation. While this is an appealing feature, the gauge color code is three dimensional and, as such, is challenging to realize with locally interacting qubits arranged on a two-dimensional surface. While some effort has been made to reduce the engineering demands of realizing three-dimensional codes via dimension jumping [50], or by finding two-dimensional variants of the gauge color code [51–53], these schemes still, respectively, either require some three-dimensional components or come at the expense of the threshold error rate.

While gauge fixing with the gauge color code offers an elegant approach to achieving a universal gate set, the space-time quantum resource cost scales equally [54–56] with other proposals, up to a constant factor, by use of single-shot error correction [54,57]. As such, despite the

apparent advantages of these codes, current proposals for magic-state distillation-based schemes with two-dimensional architectures remain attractive due to their practicality and high error thresholds [10].

## III. ENCODING QUBITS WITH THE SURFACE CODE

In this section, we review the background material that we use throughout the present paper, including stabilizer formalism [58], anyon models [1,6–9], the Clifford group, and different methods of encoding qubits with the surface code [2].

### A. Stabilizer formalism

Quantum states are robustly maintained in the code space of a quantum error-correcting code. We specify the code space of a stabilizer quantum error-correcting code with its stabilizer group $\mathcal{S}$. The stabilizer group is an Abelian subgroup of the Pauli group $\mathcal{P}$ with $-\mathbb{1} \notin \mathcal{S}$. Up to phases, the Pauli group is generated by the standard Pauli matrices, $X_j$, $Y_j$, and $Z_j$, where $j$ indicates the qubit of the system on which the operator acts.

The code space of a stabilizer code is the common $+1$ eigenspace of all the elements of the stabilizer group $s \in \mathcal{S}$, i.e.,

$$s|\psi\rangle = (+1)|\psi\rangle, \quad \forall \ s, \tag{1}$$

where $|\psi\rangle$ are basis vectors that span the code space. Stabilized states $|\psi\rangle$ are commonly known as codewords.

We act on the code space with logical operators $\bar{X}_k$, $\bar{Z}_k$, which are distinguished from other operators with bar notation. Logical operators commute with all members of the stabilizer group but are not themselves members of the stabilizer group. The logical operators generate the logical Pauli group that acts on the code space. Therefore, they satisfy the following properties: $\bar{X}_k\bar{Z}_k = -\bar{Z}_k\bar{X}_k$ and $\bar{X}_k\bar{Z}_l = \bar{Z}_l\bar{X}_k$ for $k \neq l$.

An important quantity to introduce that characterizes stabilizer codes is the code distance, which is commonly denoted $d$. The code distance is the smallest set of qubits which support one nontrivial logical operator of the code, where the support of an operator is the set of qubits an operator acts upon nontrivially. As a first-order approximation, the code distance quantifies the ability of a code to tolerate noise, as it denotes the smallest number of qubits that must be rotated in order to complete a logical operation on the code subspace of the code.

It will also be helpful to note that the action of a logical operator on the code space is invariant if the logical operator is multiplied by an element of the stabilizer group. Specifically, two logical operators $\bar{L}$ and $\bar{L}' = s\bar{L}$ for $s \in \mathcal{S}$ satisfy the relationship $\bar{L}'|\psi\rangle = \bar{L}|\psi\rangle$ for all code states $|\psi\rangle$. This result follows from Eq. (1) and the commutation relation of elements of the stabilizer group with the logical operators. It will be useful as it allows us to change, or "clean" [59], the support of logical operators, such that certain qubits of the stabilizer code do not support certain choices of logical operator.

### B. Anyons

A complementary and natural way to understand schemes of topological quantum error correction is through the language of anyonic excitations. Anyons are pointlike quasiparticles whose motion is restricted to two spatial dimensions. This restriction allows exotic exchange behavior to arise.

We will frequently invoke this quasiparticle picture to elucidate the physics of the error-correcting codes we study. For a detailed description of anyon models, we refer the reader to Refs. [1,6–8] and Appendix E of Ref. [9]. Here, we briefly review two explicit anyon models that will be relevant throughout this paper.

#### 1. $D(\mathbb{Z}_2)$ anyon model

The anyon model of the surface code is known as $D(\mathbb{Z}_2)$ [1]. It is composed of four anyons: $e$, $m$, and $\psi$, together with the vacuum particle 1, which denotes "no particle." All anyon models include the vacuum particle.

For historical reasons, the $e$ and $m$ anyons are known as the electric and magnetic charges. These anyons are their own antiparticles. This means that they will annihilate if combined. The combination of pairs of particles is captured by the notion of "fusion," denoted by the binary operation "$\times$." The fact that $e$ and $m$ excitations are their own antiparticles is captured by the fusion rules

$$e \times e = m \times m = 1.$$

Exchanging pairs of $e$ quasiparticles gives rise to bosonic exchange statistics, in the sense that they realize a trivial, i.e., $+1$, phase upon the exchange. The same is true when pairs of $m$ particles are exchanged. Braiding an $e$ and an $m$ excitation, however, has a nontrivial effect. This is seen when one particle is moved through a full loop around the other, which is known as a monodromy. Braiding an $e$ around an $m$, or vice versa, introduces a global phase of $-1$ to the system.

The $\psi$ anyon is a particle that is composed of an $e$ and an $m$ excitation, which is specified by the fusion rule

$$e \times m = \psi.$$

The particle labeled $\psi$ has fermionic exchange behavior, so its wave function acquires $-1$ global phase upon exchange. This arises from the nontrivial exchange behavior of the component $e$ and $m$ anyons.

### 2. Ising anyon model

The Ising anyon model [9,60] has two nontrivial anyon types, $\sigma$ and $\psi$. The particle $\psi$, as above, is a fermion that is its own antiparticle. The equivalence between these particles will be used in this work, so any reference to fermions refers to both interchangeably.

The $\sigma$ anyon is non-Abelian. Fusing two Ising anyons can either result in annihilation where the vacuum particle is produced, or the fusion outcome can be one fermion. This is captured by the fusion rule

$$\sigma \times \sigma = 1 + \psi.$$

The $\sigma$ particle is also able to absorb a fermion, which is represented by the fusion rule

$$\sigma \times \psi = \sigma.$$

The number of fermions absorbed by a pair of Ising anyons can be learned by fusing the pair.

The $\sigma$ anyons are equivalent to Majorana modes and can be well described by Majorana operators [9,25]. We do not require this description in this work. However, we note one important feature: The Majorana parity operator, which assigns a phase of $\pm 1$ depending on whether a pair of $\sigma$ anyons will fuse to vacuum or a $\psi$ particle, is equivalent, up to a global phase, to the operator that fuses a fermion with each $\sigma$ particle. These operators are also equivalent to a full monodromy of one $\sigma$ particle around the other. It is these operators that are used as the logical Pauli operators of qubits encoded with Ising anyons. A single exchange of two Ising anyons therefore implements a unitary rotation that corresponds to the square root of the Pauli operator. Such rotations are members of the Clifford group, which we discuss next.

### C. Clifford group

Elements of the Clifford group, $U \in \mathcal{C}$, map elements of the Pauli group onto elements of the Pauli group under conjugation. It is defined as

$$\mathcal{C} = \{U : \forall P \in \mathcal{P}, \ UPU^{\dagger} \in \mathcal{P}\}. \quad (2)$$

The Clifford group can be generated by two single-qubit unitary rotations, or "gates"—the phase gate and the Hadamard gate—which, respectively, can be expressed in terms of Pauli matrices such that

$$S = (e^{i\pi/4}\mathbb{1} + e^{-i\pi/4}Z)/\sqrt{2}, \qquad H = (X+Z)/\sqrt{2}, \quad (3)$$

together with a two-qubit controlled-NOT gate

$$\text{CNOT} = (\mathbb{1} \otimes \mathbb{1} + Z \otimes \mathbb{1} + \mathbb{1} \otimes X - Z \otimes X)/2. \quad (4)$$

The Clifford group acts on the Pauli matrices as follows. The phase gate, or "S-gate," obeys the following equations:

$$SXS^{\dagger} = -Y, \qquad SYS^{\dagger} = X, \qquad SZS^{\dagger} = Z; \quad (5)$$

the Hadamard gate, which is Hermitian, satisfies

$$HXH = Z, \qquad HYH = -Y, \qquad HZH = X; \quad (6)$$

and for the (Hermitian) controlled-NOT gate, we have

$$\text{CNOT}(X \otimes \mathbb{1})\text{CNOT} = X \otimes X,$$
$$\text{CNOT}(\mathbb{1} \otimes X)\text{CNOT} = \mathbb{1} \otimes X,$$
$$\text{CNOT}(Z \otimes \mathbb{1})\text{CNOT} = Z \otimes \mathbb{1},$$
$$\text{CNOT}(\mathbb{1} \otimes Z)\text{CNOT} = Z \otimes Z. \quad (7)$$

### D. Planar code

The planar code [2,31] is defined on an $L \times L$ square lattice with one qubit placed on each vertex of the lattice, as shown in Fig. 1. The distance of the code is $d = L$. This representation of the planar code is given in Ref. [61], but it is easily checked [62,63] that this representation, up to local unitary operations, is equivalent to the more conventional
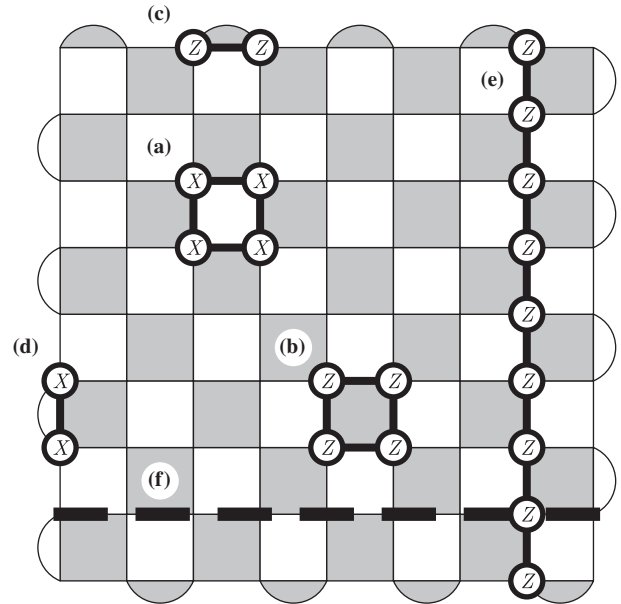


FIG. 1. The planar variant of the surface code with two rough boundaries and two smooth boundaries. Qubits lie on the vertices of the square lattice. Diagrams (a) and (b) show the two different types of weight-four stabilizer operators. Diagrams (c) and (d) show two stabilizers that lie at the boundary of the lattice. (e) The logical operator $\bar{Z}$ is the tensor product of Pauli-Z operators extending from the top to the bottom of the lattice. (f) The logical operator $\bar{X}$ is the tensor product of Pauli-X operators along the horizontal dashed line that extends from the left to the right of the lattice.

representation of the surface code [1,2] where the qubits lie on the edges of a square lattice.

To specify the stabilizer group, we bicolor the faces $f$ of the lattice black and white as in Fig. 1. With this coloring scheme, we can write the two different types of stabilizers that form the stabilizer group; namely, we have the operator

$$A_f = \prod_{j \in \partial f} X_j, \qquad (8)$$

on each white face of the lattice, and the operator

$$B_f = \prod_{j \in \partial f} Z_j, \qquad (9)$$

on each black face of the lattice. The set $\partial f$ denotes the qubits that touch face $f$. We show an example of an $A_f$ and a $B_f$ operator in Figs. 1(a) and 1(b), respectively.

We must also define stabilizers on the boundary of the planar code. The boundary stabilizers are shown in Fig. 1 by adding additional faces to the boundary of the lattice. We have added black faces to the top and the bottom of the lattice, and white faces to the left and right sides of the lattice. With the addition of these extra faces, the boundary terms are specified with the definitions given in Eqs. (8) and (9). We show explicit examples of boundary terms in Figs. 1(c) and 1(d).

Provided the stabilizer group contains only commuting generators, we have a lot of freedom as to which types of faces we can add to the boundary. Indeed, the choice of boundary stabilizers plays a very important role in the encoding properties of the planar code (see Ref. [31]). To maintain consistency with the terminology used in Refs. [2,31], we call boundaries with black faces on the boundary "rough boundaries" and boundaries with white faces on the boundary "smooth boundaries." In Fig. 1, we show rough boundaries on the top and bottom of the lattice and smooth boundaries on the left and right sides of the lattice.

With the choice of boundaries shown in Fig. 1, the planar code encodes one logical qubit. The logical operators are strings of Pauli operators that extend between different boundaries of the same type. The logical operator $\bar{Z}$ is the tensor product of Pauli-$Z$ operators that are supported along a string that extends from the top to the bottom of the lattice, between two rough boundaries. We show $\bar{Z}$ in Fig. 1(e). The logical operator $\bar{X}$ is the tensor product of Pauli-$X$ operators supported on a string that runs from the left side to the right side of the lattice. The qubits that support $\bar{X}$ lie on the thick dashed line shown in Fig. 1(f). Unlike $\bar{Z}$, the operator $\bar{X}$ is a string that stretches between two different smooth boundaries of the lattice. It is easily seen that $\bar{X}$ anticommutes with $\bar{Z}$, as they are commonly supported on only one qubit of the lattice.

It is worth noting that, provided the string operators commute with the stabilizer group, we are free to choose logical operators along any string on the lattice as long as the terminal boundaries of the string are not changed. This is easily seen by observing that we can manipulate the path of the strings of logical operators shown in Fig. 1 by multiplying them by elements of the stabilizer group. We frequently make use of this fact in discussions in later sections of the present paper.

The logical operators of the planar code can be understood using the picture of anyonic quasiparticle excitations. Indeed, as it is explained in Ref. [1], stringlike Pauli operators acting on the codewords of the planar code can be regarded as creation operators for anyons. Specifically, a string operator of Pauli-$Z$ (Pauli-$X$) operators with end points that lie in the bulk of the lattice create pairs of electric (magnetic) charges at the end points of the strings, as we have defined in Sec. III B 1. Hopping operators for $e$ and $m$ excitations also correspond to stringlike operators.

Unlike the creation and hopping operators we have just mentioned, the stringlike logical operators of the planar code do not create quasiparticle excitations. This is because the string operators terminate at boundaries of the planar code where quasiparticle excitations are absorbed, or "condensed." Specifically, a rough (smooth) boundary is capable of absorbing $e$ ($m$) particles. We can therefore regard the $\bar{Z}(\bar{X})$ logical operator as the process of creating a single $e$ ($m$) excitation at one rough (smooth) boundary and subsequently transporting the particle across the lattice where it is then absorbed by the opposite rough (smooth) boundary. The braiding statistics of the exchange of these two excitations assures the appropriate commutation relations between these logical operators. We will frequently draw on this picture to demonstrate different logical operations using the surface code. Our freedom to deform logical operator strings is also elucidated in this picture, as we have argued that it is the role of logical operators to transport excitations between different boundaries. For the case of the planar code, the action of the operator is independent of its path along the lattice, provided charges are transported between the appropriate boundaries. In general, the action of logical string operators are invariant under continuous deformations to their path over the lattice.

### E. Encoding logical qubits using holes

We can increase the number of encoded qubits for the surface code by introducing punctures to the lattice. We show a puncture, or hole, in Fig. 2. The qubits inside the puncture, shown as small black points, have been disentangled from the lattice. We point out that unlike the lattice shown in Fig. 1, the punctured lattice has only rough boundaries. As such, without a hole in the center, this lattice would encode no logical qubits. By introducing a puncture in the center of the lattice, we are able to encode a single logical qubit on the lattice. We show the logical operators for the encoded qubit in Figs. 2(a)
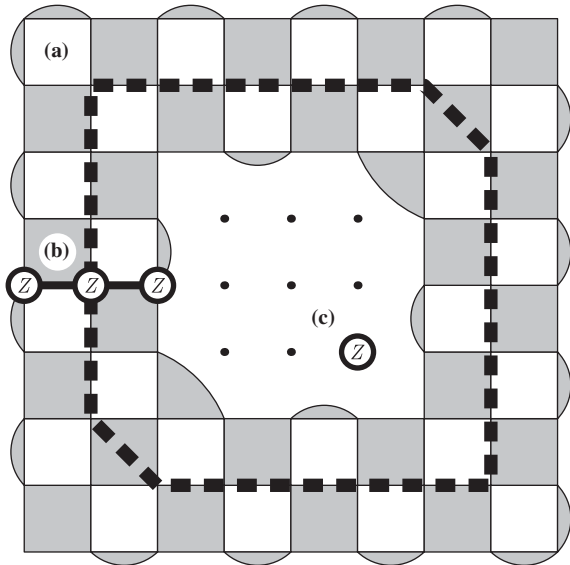
FIG. 2. A hole in the planar code modifies its code space. A hole is shown in the center of the lattice. (a) When a hole is prepared, a new logical operator is produced which consists of a nontrivial cycle of Pauli-$X$ operators that enclose the hole. The qubits that support the logical operator are marked by a thick dashed line. (b) The logical operator shown by label (a) anticommutes with the conjugate logical operator which is composed of the tensor product of Pauli-$Z$ operators that extends from the boundary of the hole to the boundary of the surface. (c) A hole can be produced in the lattice by measuring the qubits in the computational, i.e., Pauli-$Z$, basis.

and 2(b). The logical operator $\bar{X}$ is the tensor product of Pauli-$X$ operators supported on a cycle of qubits that enclose the puncture shown by a dashed line in the figure. The logical operator $\bar{Z}$ is the tensor product of Pauli-$Z$ operators supported along a line that connects the lattice boundary to the boundary of the puncture. We show such an operator in Fig. 2(b). As with the planar code, the logical operators here are stringlike and allow continuous deformations.

Like the planar code, the logical operators of the punctured planar code can also be interpreted from the point of view of topological excitations. Given that both the lattice boundary and the hole in the lattice have rough boundaries, they are both capable of absorbing an $e$ particle. Therefore, the logical state of the hole relates to the number of electric charges that have been passed from the lattice boundary into the hole. Specifically, the parity of this number gives one two-level logical degree of freedom. The even case corresponds to vacuum due to the fusion rule $e \times e = 1$, and the odd case corresponds to an $e$ charge absorbed by the hole. We measure the charge parity absorbed by the hole with a measurement that encloses the hole, which is able to count the number of pairs of electric excitations that have been shared between the lattice boundary and the hole, thus giving the topological structure of the $\bar{X}$ logical operator. Indeed, this measurement is analogous to Gauss' law [1].

The distance of a code where qubits are encoded using holes depends on two quantities: the length of the boundary of a given hole and the separation of a hole from another lattice boundary. As such, to maintain a code distance $d$, all holes with a common boundary type must be separated by a distance of at least $d$ and must maintain a distance of $d$ from the boundary of the lattice. Holes must also have a boundary of at least length $d$. As such, for the case of spherical holes, i.e., holes where the removed qubits are taken from a simply connected region, the radius of the region must be about $d$. For simplicity, we only consider holes that cover simply connected regions.

In general, we can consider punctures with smooth boundaries instead of rough boundaries, where smooth boundaries absorb $m$ particles instead of $e$ particles. Indeed, protocols to implement quantum logic gates have been shown which involve holes with both types of boundaries [10–12]. It is also worth noting the recent work of Delfosse et al. [64], where they study the encoding properties of holes that have both rough and smooth boundaries. Here, we need only introduce holes with rough boundaries. In what follows, we will see that rough boundaries differ from smooth boundaries only by lattice defects that are introduced in the following subsection.

### F. Encoding logical qubits with twist defects

We next consider encoding qubits using twist defects. Twist defects were introduced for the surface-code model by Bombín in Ref. [20]. Other work on twist defects on the surface code is presented in Refs. [25,33,37,65,66]. Twist defects have also been introduced in the color-code models in Ref. [21,67], and from the point of view of topological quantum field theories in Refs. [22,23] and references therein. See also the second item in Sec. 12 of Ref. [9].

In Fig. 3, we show four twists on a lattice, which collectively encode a single qubit. On this lattice, the qubits that lie along the two pink lines are removed from the lattice. We refer to these pink lines as defect lines. Twist defects lie on the double plaquettes where the defect lines terminate. Up to a complex phase, we show a stabilizer that lies on a twist defect in Fig. 3(a). Indeed, one should also include a complex $\pm i$ phase in front of the twist stabilizer such that the stabilizer returns a $+1$ measurement outcome, where the sign determines the phase the defect carries. We largely neglect this phase, as it does not significantly change the physics that we are demonstrating, but we direct the interested reader to Ref. [20]. In Fig. 3(b) we also show how the weight-four stabilizers are modified along the defect line. In general, the stabilizers that lie on the defect lines are determined as follows: We first replace the stabilizers that lie along defect lines with restricted stabilizers, where we take the restriction [68] on the qubits that do not support defect lines. We then replace the restricted stabilizers with the products of pairs of adjacent restricted stabilizers. Indeed, following this prescription gives the
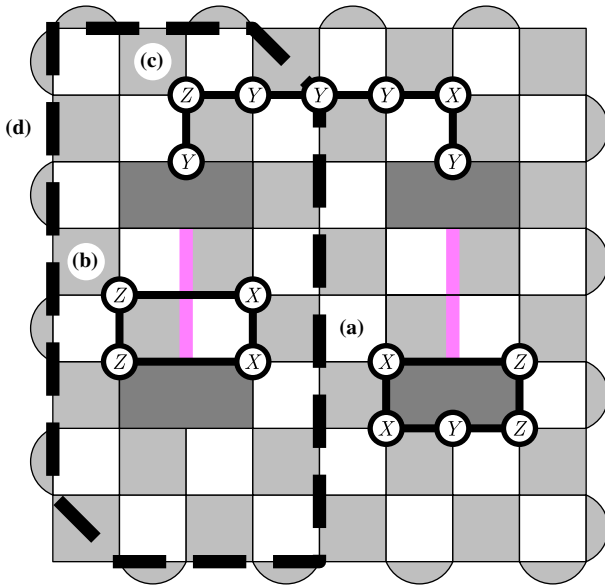
FIG. 3. Two pairs of twist defects on the planar code. Twists appear at the end points of the pink defect lines. (a) The weight-five stabilizer operator of a twist defect, shown up to a phase factor. (b) A modified weight-four stabilizer that lies on the pink line. (c) A logical operator associated with the four twist defects on the lattice. (d) The tensor product of Pauli-$X$ operators on the qubits supported along the dashed line gives the logical operator that anticommutes with that shown at (c).

stabilizers shown in Figs. 3(a) and 3(b). We show the logical operator $\bar{X}$ in Fig. 3(c), and the logical operator $\bar{Z}$ is the tensor product of Pauli-$X$ operators on all the qubits that support the dashed line in Fig. 3(d). If we have a single uniform lattice boundary, we can see from the logical operators that to maintain a distance $d$ code, all twist defects must be separated by at least $d$ (see Ref. [24]).

We now briefly review the properties of twists from the point of view of anyonic quasiparticles. Indeed, the string operators we have studied create either a pair of electric charges or magnetic charges at the two end points of the string when they act on code words in the bulk of the lattice. In this sense, we see that the surface-code model obeys global charge-parity conservation law, as anyonic charges of the same type must be created in pairs. Interestingly, string operators that cross a defect line create one $e$ excitation and one $m$ excitation at its end points. Remarkably, it follows from this fact that this property gives twist defects the ability to absorb a fermion. It was observed by Bombín in Ref. [20] that this behavior is reminiscent of Ising anyons, which we reviewed in Sec. III B 2. Importantly, like twists, Ising anyons also have the ability to absorb fermionic excitations.

We can see that twist defects mimic the behavior of Ising anyons by consideration of their logical operators. Indeed, the string of Pauli-$Y$ operators shown in Fig. 3(c) represents a string operator that transports a fermion from one twist on the lattice to another. The operator $\bar{Z}$, on the other hand,

measures the fermionic charge parity that has been absorbed by the two twist defects that lie on the left of the lattice. With this observation, we see that the physics of Ising anyons is echoed by the simple stabilizer model that we consider here. Later, we make use of this analogy to find new logical gates by code deformation, which we discuss in the following section.

## G. Twist on the boundary

Before moving on to the next section, we finally demonstrate a correspondence between twist defects and the corners of the planar code above. More precisely, by corners, we mean points on the boundary of the planar code where rough boundaries meet smooth boundaries. In later sections, we make use of this correspondence to perform Clifford gates with the planar code using code deformations.

To show the correspondence between twists and corners, we consider the planar-code lattice with defect lines in different configurations. To do so, we first build a diagrammatic notation without the microscopic details of the underlying lattice.

In Fig. 4, we show a planar code with a rough boundary that supports a single pair of twist defects. In the diagram, strings of Pauli-$Z$ (Pauli-$X$) matrices are shown by red (blue) strings. With this picture, the operator $A_f$ ($B_f$) can be regarded as a red (blue) string that follows a trivial cycle, as shown in Figs. 4(a) and 4(b), respectively. As the lattice has only a single rough boundary, the stabilizer group also contains strings of Pauli-$Z$ operators that follow trivial cycles and terminate at the boundary of the lattice, as shown
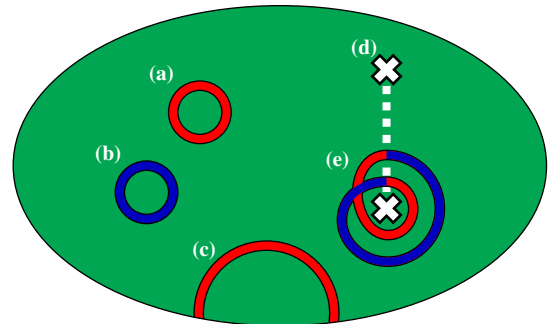


FIG. 4. A schematic diagram of the planar-code lattice with a rough boundary. Strings of Pauli-$Z$ (Pauli-$X$) operators are marked by red (blue) lines, respectively. Diagrams (a) and (b), respectively, show $A_f$ and $B_f$ stabilizer operators. (c) An element of the stabilizer group where a string of Pauli-$Z$ operators follows a trivial path and terminates at the same rough boundary of the lattice. (d) A defect line marked by a dashed white line with two twist defects at its end points. Red strings that cross the defect line change to blue and vice versa. (e) A stabilizer operator that encloses a single twist. The string must wind twice around the twist, which changes the string from red to blue and back to red again to form a closed loop. As mentioned, the stabilizer should also include a complex phase due to the point where the two strings cross, which we do not consider explicitly.

in Fig. 4(c). Importantly, as the boundary is entirely rough, only red strings can terminate at the boundary. We show a defect line in Fig. 4(d), marked by a thick dashed white line, where two twist defects lie at its termination points. Twists are shown as white crosses. We also show an example of a stabilizer operator that encloses a single twist in Fig. 4(e) drawn as strings of Pauli-$X$ and Pauli-$Z$ operators, where the string changes color as it crosses the boundary.

Having introduced a simple diagrammatic notation, we can easily demonstrate the equivalence between the corners of the planar code and twists, which we summarize in Fig. 5. In Fig. 5(a), we show a lattice that encodes one logical qubit using four twist defects. The logical operator $\bar{X}$ is a string operator that encloses the top two twists. As previously discussed, this operator corresponds to an operator that transports a fermion between the two enclosed twists. We see this because the parallel strings are passing an $e$ excitation and an $m$ excitation between the two twists. The operator $\bar{Z}$ is shown by a red loop that encloses the two twists to the left of the figure. This measures the parity of fermionic charges that have been absorbed by the two enclosed twists. From the diagram, we can see that the two logical operators anticommute because there is one single point where the blue segment of $\bar{X}$ intersects the red $\bar{Z}$ loop.
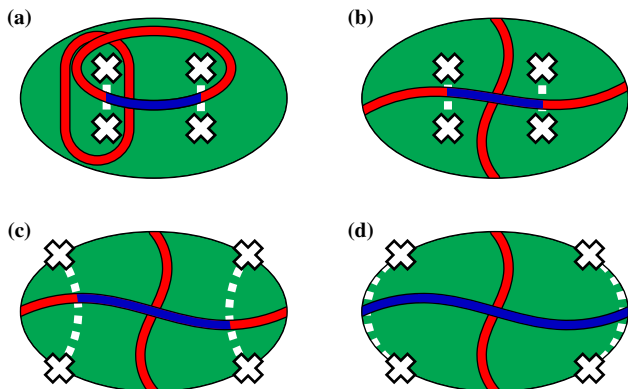


FIG. 5. Lattices with different configurations of twist defects but equivalent logical operators. (a) Four twist defects in the center of the lattice. The logical operators are closed loops that enclose pairs of twists. (b) The logical operators shown in (a) multiplied by stabilizer operators that terminate at the boundary, as in Fig. 4(c), thus giving the logical operator shown in the picture. (c) We consider the lattice shown in (a) and (b), except where the defect lines terminate at the boundary. In this picture, the twist defects lie at the boundaries of the lattice. Importantly, the logical operators are unchanged from those shown in (b). (d) We finally show the lattice where the defect lines are drawn close to the left and right sides of the lattice. Now, the logical operators are equivalent to those in Fig. 1, where a string of Pauli-$X$ operators runs from the left to the right of the lattice and a string of Pauli-$Z$ operators runs from the top to the bottom of the lattice.

Logical operators can be manipulated by multiplying them by stabilizer operators. As we have discussed, stabilizer operators are strings of trivial cycles or strings that both terminate at the same boundary. In Fig. 5(b), we show the logical operators of Fig. 5(a), where the logical operators are deformed such that they terminate at the boundary of the lattice. As with Fig. 5(a), we see that these two logical operators anticommute, as there is still a single point where the blue segment of the $\bar{X}$ operator intersects the red string that represents the $\bar{Z}$ operator.

Indeed, the logical operators in Fig. 5(b) are reminiscent of the logical operators of the planar code discussed in Sec. III D, where logical operators also terminate at the boundary. We next consider the same model, except where the defect lines terminate at the boundary of the lattice, such that the four twist defects of the code are located at the boundaries of the lattice. This is shown in Fig. 5(c). In this picture, the logical operators are unchanged from those shown in Fig. 5(b), which, once again, are logical operators that are very similar to those of a planar code. To make this analogy completely clear, we show the same model in Fig. 5(d), except where the defect lines have been drawn close to the boundary of the lattice but where the twist defects remain in the same locations. In this picture, the logical operators are now identical to those of the planar code, where $\bar{Z}$ corresponds to moving an $e$ excitation from the upper boundary to the lower boundary, and $\bar{X}$ corresponds to moving an $m$ excitation from the left boundary to the right boundary. In this sense, we can regard a smooth boundary as a rough boundary that has been covered by a defect line. We also observe that the points, or corners, where the rough boundary meets the smooth boundary are the locations of twist defects. We can therefore regard the corners of the planar code as equivalent to twist defects. In the following section, we make use of this analogy to demonstrate how we can perform logical Clifford gates on the planar code by code deformations that manipulate the corners of the planar-code lattice as though they are twist defects.

## IV. LOGICAL OPERATIONS BY MANIPULATING CORNERS

We have now introduced several different methods of encoding qubits in the surface-code model. However, quantum computation requires that we also perform logical gates on encoded qubits. A well-studied method for performing fault-tolerant quantum logical operations is by use of code deformations [10–19], where we make special measurements to manipulate the logical qubits of a quantum error-correcting code. In particular, we perform code deformations to manipulate and braid the twist defects that lie at the corners of the planar code, as discussed in the previous section. Given that the twist defects of the planar code are analogous to Ising anyons, we can devise code-deformation strategies based on known braiding

gates that can be achieved using Ising anyons [9,69]. The manipulation of twist defects has been discussed in Refs. [20,21,25,67,70]. Indeed, like Ising anyons, twist defects can achieve the Clifford group by braiding, which, together with magic-state distillation [43], can be used to achieve universal quantum computation. In this section, we begin by reviewing the theory behind code deformations. We then show explicitly how to move twists by code deformations. Finally, at the end of this section, we show how to perform single-qubit Clifford gates.

## A. Code deformations

Here, we briefly review the concept of code deformation. This is a process by which a code defined with stabilizer group $\mathcal{S}$ is mapped onto a different code defined by stabilizer group $\mathcal{S}'$. With a suitable choice of code deformation, we can perform logical rotations on a code. Code deformation is achieved simply by measuring the elements of $\mathcal{S}'$ to project onto the new code. Details on performing measurements using the stabilizer formalism are given in Chapter 10 of Ref. [71]. We note that the code-deformation procedures we present should be readily adapted to adiabatic topological quantum-computational schemes using the methods shown in Refs. [72,73].

We require that no stabilizers of $\mathcal{S}'$ measure the logical information encoded in the code space of the code specified by $\mathcal{S}$ to preserve encoded information. For all of the instances we consider here, we can perform sequential measurements with weight that is much smaller than the distance of the code. It then follows that we can always clean logical operators [59,74] away from the qubits that support the stabilizer measurements of $\mathcal{S}'$. We therefore focus on how the stabilizer group is modified by the stabilizer measurements of $\mathcal{S}'$ on the code specified by $\mathcal{S}$.

We consider the simple case where the generators of $\mathcal{S}'$ differ from $\mathcal{S}$ by only a single element, $s'$. This is easily generalized to the case where $\mathcal{S}'$ differs from $\mathcal{S}$ by many elements, as we can sequentially deform the code several times between many different codes. Depending on the choice of $\mathcal{S}'$, element $s'$ will do one of two things: It will either commute with all elements of $\mathcal{S}$, or it will anticommute some stabilizers of $\mathcal{S}$. In the case where $s'$ commutes with all elements of $\mathcal{S}$, provided $s'$ is not a logical operator, it must follow that $s' \in \mathcal{S}$, which is a trivial deformation.

We next consider the case where $s'$ anticommutes with some elements of $\mathcal{S}$. We denote the subset of elements of $\mathcal{S}$ that do not commute with $s'$ as $\mathcal{A}$, where we denote members of the anticommuting set $s_j \in \mathcal{A}$, where $1 \le j \le N$ and $N$ is the number of elements in $\mathcal{A}$. In this case, we replace elements of $\mathcal{A} \subseteq \mathcal{S}$ with terms of the form $s_j s_{j+1}$ for all $1 \le j \le N-1$. Then, up to the measurement outcome $m' = \pm 1$, we project our encoded state onto the code space of $\mathcal{S}'$. In the case where $m' = -1$, we project the encoded state onto the $-1$ eigenstate of $s'$. Then, we can apply a

unitary correction operator to rotate the state we achieved under the projection onto the desired state.

## B. Manipulating corners by code deformations

We now show that we can move the corners of the planar code onto the bulk of the lattice. The sequence of measurements we choose is based on the terms in the perturbation expansion used to synthetically introduce twist defects to Hamiltonian models [75].

In Fig. 6(a), we show the single-qubit Pauli measurements that we perform along the pink line to move the twist into the bulk of the lattice. Upon performing these measurements, the twist defect is moved along the pink line. After performing these measurements, the qubits that lie on the pink line are projected onto a product state and are thus disentangled from the lattice.

With a few exceptions, which we show in Fig. 6, the stabilizers along the pink defect line of the deformed lattice are found by the same prescription given in Sec. III F. Twist defects are weight-five stabilizers at the terminal point of the defect line, and stabilizers along the straight segments of the defect line are weight-four operators that straddle the line. We also consider how the stabilizers are modified in
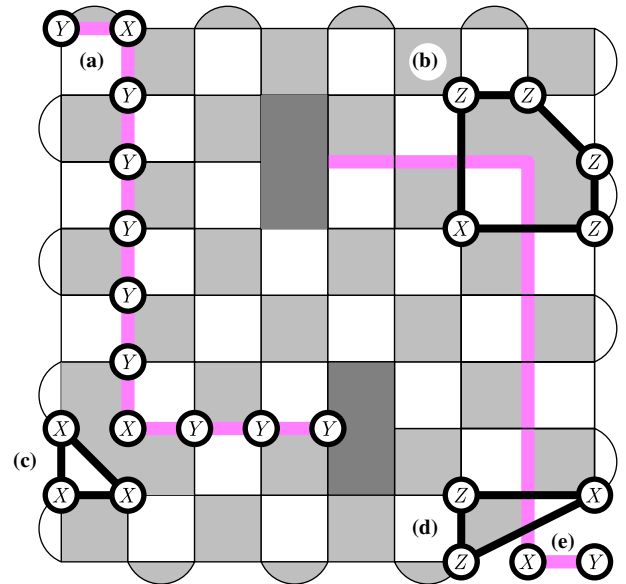


FIG. 6. Moving the twist defects at the corners of the planar code into the bulk of the lattice. (a) We make single-qubit Pauli measurements along the pink defect line to deform the twist defect from the corner of the planar code to the center of the lattice. Most of the measurements along the line are Pauli-$Y$ operators, but we also use Pauli-$X$ measurements and Pauli-$Z$ measurements to move a twist around a corner. (b,c) Stabilizer operators that are modified by the code deformation at locations where twists are moved around corners. (d) The stabilizer operator that lies at the corner of the lattice where a twist defect has been removed. (e) Two qubits at the corner of the lattice that have been measured in a product basis and are no longer entangled with the code.

the locations where the defect line moves around a corner. In Figs. 6(b) and 6(c), we show some explicit examples of one weight-three stabilizer, and one weight-five stabilizer where the twist defect turns around a corner on the lattice. We also show how the stabilizers are modified close to the corner where the twist defect originated in Figs. 6(d) and 6(e). In Fig. 6(d), we show a weight-three stabilizer on the boundary of the lattice where the twist defect began, and in Fig. 6(e), we show two qubits that have been projected into the product state and are thus removed from the code.

## C. Single-qubit Clifford rotations on the planar code by code deformation

Having demonstrated that it is possible to move the corners of planar codes into the bulk of the lattice and braid them like twist defects, we now show that we can complete the generating set of the Clifford group using twist defects. In terms of scope and resource usage, our scheme can be compared with protocols given in Refs. [16,76] where Hadamard gates are performed on qubits encoded with pairs of holes.

In Fig. 7, we show that a deformation that exchanges two twists on two adjacent corners maps between different Pauli matrices. Specifically, we show that exchanging the two twists at the bottom of the lattice will, up to phases, exchange logical operators $\bar{Z} \rightarrow \bar{Y}$ and $\bar{Y} \rightarrow \bar{Z}$, where $\bar{X}$ remains invariant. We work through this manipulation, step by step, by manipulating the support of the logical operators shown in Sec. III G. The strategy we follow is to deform the logical operator away from the path of the twist defects.

Note that, up to phases, the effect of this exchange is equivalent to the braiding of two Ising anyons [69,77]. Indeed, as noted in Sec. III B, exchanging two Ising anyons is equivalent to the square root of the $\bar{X}$ operation associated with these twists.

We first deform the two twists on the corners of the lower boundary into the center of the lattice, as shown in Fig. 7(a). Upon doing this, we deform the logical operator $\bar{Z}$ away from the positions of the twists by multiplying the logical operator by elements of the stabilizer group such that none of the qubits that are measured during the code deformation supports the logical operator. The logical $\bar{Z}$ operator is shown in red in Fig. 7(a), snaking between the twists in the center of the lattice.

Next, we continue to move the twist defect that began in the bottom-left corner of the lattice into the bottom-right corner of the lattice. We therefore continue to deform the logical operator such that one end point terminates on the right-hand side of the lattice. This forces the logical operator to move over a defect line which has already been drawn on the lattice, which changes its string type from red to blue, as we show in Fig. 7(b). We also multiply the logical operator by an element of the stabilizer group which extends from the left boundary to the lower
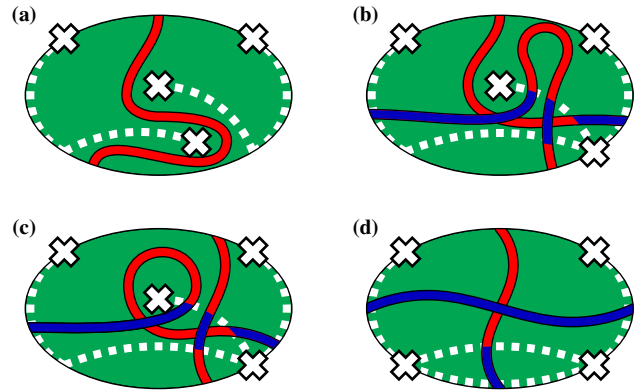


FIG. 7. A sequence of code deformations that map the logical operator $\bar{Z}$ onto $\bar{Y}$. (a) We move the bottom two twist defects into the center of the lattice. We also show the $\bar{Z}$ string operator in red, which "snakes" between the twist defects at the center of the lattice. (b) We deform the position of one twist defect into the bottom-right corner of the lattice. The logical operator is deformed over the defect line such that it is not supported on any of the qubits where code-deformation measurements move the lowest twist defect from its position in (a) to its position in (b). This deformation moves the logical operator over a defect line and, as such, changes the color of the string. We also multiply the logical operator by a stabilizer that extends from the left-hand side of the lattice to the bottom of the lattice, which is also shown in the figure. (c) A small deformation of the logical operator shown in (b) gives the pictured logical operator. (d) Multiplying the logical operator shown in (c) by a stabilizer that encloses the single twist defect lying at the center of the lattice deforms the logical operator out of the path of the twist, enabling us to move the twist defect at the center of the lattice to the bottom-left corner of the lattice, thus completing the procedure.

boundary of the lattice. Introducing this stabilizer operator at this point will help us see that we can deform the logical operator out of the trajectory of the twist defect that lies in the center of the code as it moves towards the bottom corner of the lattice.

In Fig. 7(c), we show a small deformation of the logical operator shown in Fig. 7(b), where now we have the product of a string operator running from the top to the bottom of the lattice, and a string that runs horizontally across the lattice that loops around the one twist defect that remains in the bulk of the lattice. To deform the logical string shown in Fig. 7(c), we multiply the logical operator by a stabilizer that loops around a single twist, as in Fig. 4(e), to deform the horizontal string over the central twist. This method allows us to move the central twist defect into the bottom-left corner of the lattice without performing any code-deformation measurements over the support of the logical operator. Then, recognizing that the two parallel defect lines that run along the bottom edge of the lattice are equivalent to a boundary where there is no defect line, we recover the lattice shown in Fig. 5(d).

We finally look at the action of the deformation on the $\bar{X}$ operator. One can readily check that this logical operator is invariant under the presented transformation, as

it can be supported on the qubits on the lattice that are never acted upon by nontrivial measurements under the code-deformation procedure, thus showing the promised action of this code-deformation procedure.

To complete the Clifford group, one can also check using a similar argument that, up to phases, exchanging the two twists on the left-hand side of the lattice will map the Pauli operators such that $\bar{X} \to \bar{Y}$, $\bar{Y} \to \bar{X}$, and it will leave the operator $\bar{Z}$ unchanged. For an alternative perspective, we show this operation in a spacetime diagram in Fig. 8. Now, if we denote the exchange of the two twists at the bottom (left-hand side) of the lattice as $B_1$ ($B_2$), then, up to Pauli rotations, we have the single-qubit logical phase gate $\bar{S}$ and the logical Hadamard gate $\bar{H}$ such that $B_2 = -\bar{X}\bar{S}$ and $B_1 B_2 B_1 = -\bar{Y}\bar{H}$, which generate the single-qubit gates of the Clifford group. Given that we can achieve logical Pauli matrices either via transversal single-qubit Pauli rotations or, more simply, by updating the Pauli frame [16], we recover a fault-tolerant implementation of the Clifford group with code deformation using the planar code.
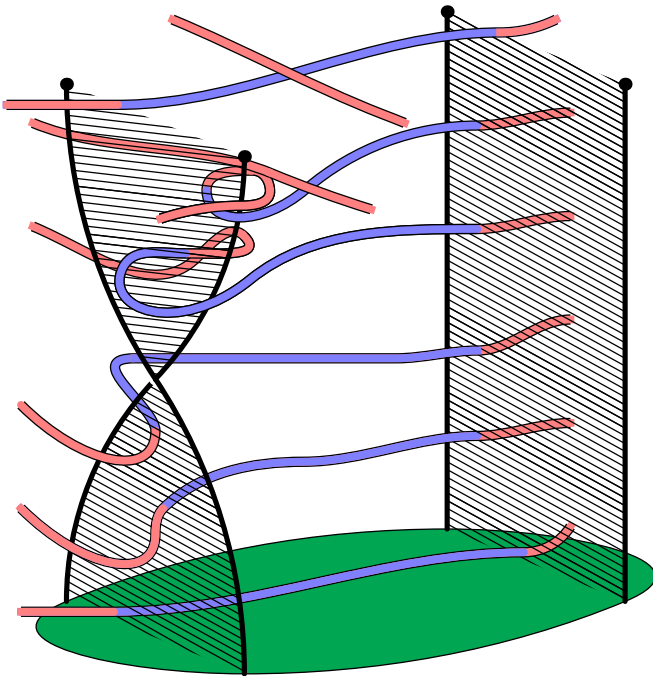
Upon performing the suggested braiding operations with the square lattice introduced in the previous section, we modify the code distance. Recalling that the distance of a logical qubit encoded via twist defects depends on the separation between all the twists on the lattice, it is obvious that we must deform the locations of the twist defects such that they all maintain a large separation. Using the lattice shown in Fig. 1, we can find paths that the twists follow while undergoing this exchange such that all the twist defects maintain a distance of at least $L/2$ from one another on the considered lattice geometry. As such, the code distance is $d \sim \mathcal{O}(L/2)$ as we perform the presented logical operations. Interestingly, we also find that for the code with a rotated geometry, as is shown in Fig. 9, we can exchange the corners without more than a small constant loss in code distance. This comes at the expense of using $2d^2$ qubits to achieve a code of distance $d$, in contrast to the square geometry we have already introduced, which requires only $d^2$ qubits. It is interesting then that the resource demands of both lattice geometries are similar when we consider performing logical gates by braiding the corners of the lattice.



FIG. 8.    Spacetime figure showing the evolution of the logical operator $\bar{X}$ under the exchange of two twists, where the time direction is upwards along the page. The figure shows that the operator is mapped onto the logical operator $\bar{Y}$, which can be seen at the top of the figure. The trajectory of the twist defects is marked by black lines. As in the two-dimensional figures above, red (blue) strings correspond to the world lines of $e$ ($m$) quasiparticle excitations. We continue with the convention we used earlier, where $e$ charges can terminate at the boundary. We depict this by showing red strings diverging away from the twist defects.
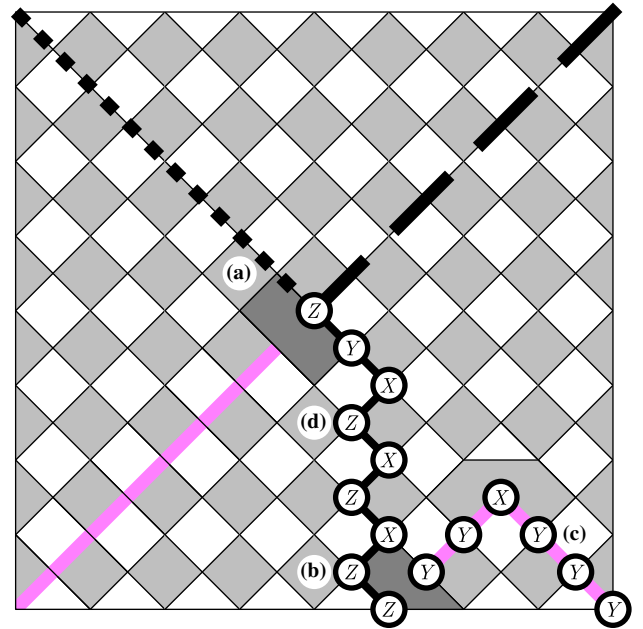


FIG. 9.    By rotating the lattice geometry about an angle of $\pi/4$, we can perform both twist exchange operations without reducing the distance beyond a small constant value. The figure shows a distance $L = 9$ lattice. (a,b) Two twist defects that have been deformed into the center of the lattice. (c) The single-qubit measurements we make to move the twist along the bottom of the lattice. (d) A weight-nine logical operator at a moment where the two twists pass each other. The black dashed and dotted lines show the support of conjugate weight-nine logical operators that we find to be minimal in weight.

### D. Encoding two qubits on a single planar code

By regarding the corners of the planar code as twist defects, we have seen that we can braid them to realize a set of gates that generate the single-qubit elements of the Clifford group. We can extend this idea by adding more corners to the planar code, such that it encodes more qubits. Then, we can braid the corners of this extended code to implement entangling gates, as we describe in this subsection. We believe that the entangling operation we present here may provide a relatively simple procedure to entangle two fault-tolerant qubits in the laboratory using near-future technology.

We look at the example shown in Fig. 10, which encodes two logical qubits that we index qubit 1 and qubit 2. In the picture, we show a planar code with six twist defects lying on the boundary of the surface code. The two twist defects at the left of the figure and the twist defect at the bottom center of the lattice support the logical qubit 1, and the other three twist defects support qubit 2. The logical operators, $\bar{X}_1$, $\bar{Z}_1$, $\bar{X}_2$, and $\bar{Z}_2$, are shown in the figure. As in the previous subsection, we can generate the single-qubit Clifford operations by exchanging the twist defects of a respective qubit.

Additionally, we can also perform an entangling operation between the two qubits by exchanging the two twists in the center of the lattice.

Each pair of twists is individually able to fuse to either vacuum of a $\psi$ particle. However, for any state within the stabilizer space, the parity of the number of $\psi$ particles must be even overall. This even parity is to ensure that they can mutually fuse to vacuum. Given this restriction, the fusion result for the leftmost and rightmost pairs must always be the same as that of the central pair. This pair is
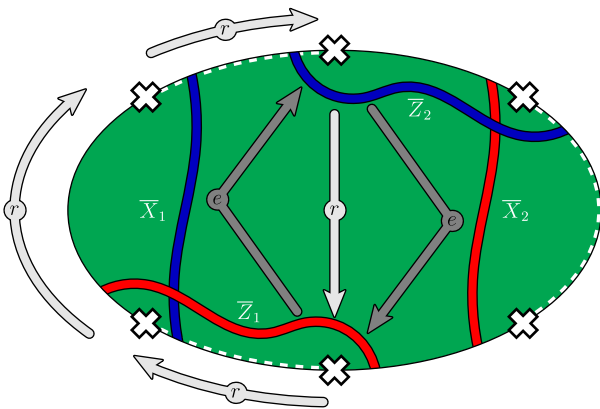


FIG. 10. A surface code that encodes two logical qubits. Logical operators $\bar{X}_1$, $\bar{Z}_1$, $\bar{X}_2$, and $\bar{Z}_2$, which act on qubit 1 and qubit 2, are shown. Exchanging the two central qubits along the trajectories shown by the dark grey arrows, which are marked by $e$, entangle the two logical qubits. This exchange can be performed without decreasing the code distance beyond a small constant amount on an $L \times 2L$ lattice with the square geometry we show in Fig. 1.

therefore associated with the logical operator $\bar{X}_1 \bar{X}_2$. Consider exchanging these two twists along the dark grey arrows shown in Fig. 10, which are marked with a letter $e$. Explicitly, we find that, under the exchange of the central two twists, we get the transformation

$$\bar{X}_1 \to \bar{X}_1, \qquad \bar{Z}_1 \to \bar{Y}_1 \bar{X}_2,$$
$$\bar{X}_2 \to \bar{X}_2, \qquad \bar{Z}_2 \to \bar{X}_1 \bar{Y}_2,$$

which can be shown using methods similar to those used in Sec. III B. This gate is equivalent to a controlled-NOT gate up to local Clifford rotations.

Remarkably, we can perform both the single-qubit operations and the entangling gate, without decreasing the distance of the code, by using an $L \times 2L$ lattice with square geometry as shown in Fig. 9, where we have about $2d^2$ physical qubits describing two logical qubits with a distance of about $d$ code. As such, we believe, given the recent surge in progress in experimental quantum error correction [78–85], that this example provides a relatively simple experiment to demonstrate the full Clifford group that may be implemented in the near future.

Note also that by exchanging the leftmost and middle twists using the light grey arrows marked $r$, and similarly for the right-hand side, we can achieve the direct exchange of any twist pair in the center without loss of distance. The equivalent gates can be achieved without using such rotations, but it could yield superior results in certain cases by keeping the twists slightly better separated.

## V. ENTANGLING DIFFERENT TYPES OF LOGICAL QUBITS

In addition to performing single-qubit elements of the Clifford group, it is also important to have fault-tolerant schemes that entangle many logical qubits. It is known that we can perform entangling operations using two-qubit nondestructive parity measurements [4]. One method of performing a two-qubit parity measurement nondestructively is to prepare a third ancillary qubit which is then entangled to the two qubits we wish to perform a parity measurement over, and to then measure the ancilla qubit. In this section, we elaborate upon one of the fault-tolerant entangling schemes in Ref. [24], which performs a controlled-NOT gate between two qubits that are encoded using quadruples of twists using a third ancillary logical qubit. We first show in detail that we can perform a controlled-NOT gate between a logical qubit encoded with a pair of holes and a logical qubit encoded over four twist defects by code deformations. We can then use this entangling gate to perform parity measurements between qubits encoded with twists by performing entangling gates between logical qubits and an ancillary qubit encoded with a pair of holes, and subsequently measuring the ancilla qubit. We use this entangling operation to perform a

controlled-NOT gate between qubits encoded with twist defects. We point out that this scheme is much like a measurement-only topological quantum computation scheme [39,40,65,69,86–88] presented by Bravyi in Ref. [69].

## A. Braiding twists and holes

We consider the setup shown in Fig. 11. In the picture, we show two qubits: one encoded over four twists on the lattice and the other encoded using two punctures. We call a qubit encoded with four twists the quadruple-twist encoding, and a qubit encoded using two punctures the hole-pair encoding. We show that deforming one hole along a path that encloses two twists, whose trajectory we show with a dashed line in the figure, will execute a controlled-NOT gate. We show the action of the braid operation by demonstrating that the logical operators shown in Fig. 11 satisfy the conditions given in Eqs. (7).

We will not elaborate in detail on the sequence of code-deformation measurements that we perform to move holes, as this has been described in detail in, for instance, Refs. [10–16]. For now, it is enough to understand that holes are moved by first increasing their size along some suitable direction and then decreasing their size again, such that the hole is displaced along their path of motion. More specifically, we increase the size of a hole with a rough (smooth) boundary by measuring the physical qubits of the lattice with single-qubit Pauli-$Z$ (Pauli-$X$) close to the boundary of the puncture. As such, the qubits in the center of a hole with a rough (smooth) boundary are measured onto an eigenstate of the Pauli-$Z$ (Pauli-$X$) matrix [see Fig. 2(c)]. To decrease the size of the hole, we measure stabilizers at the boundary of the puncture. Importantly, as we perform measurements to move the hole, we will not change its topology. We can continually repeat this prescription to transport a hole along the different trajectories described below.

Having discussed how holes are moved around the lattice, we next show that the braid shown in Fig. 11 will result in a controlled-NOT gate. We first show that $\bar{Z}_T \rightarrow \bar{Z}_C \bar{Z}_T$. This is shown diagrammatically in Fig. 12. We consider the transformation of the $\bar{Z}_T$ that is shown in Fig. 11. As the hole moves, the logical operator stretches to follow the hole. This causes the logical operator to thread between the twists, as we show in Fig. 12(a). We note that as the hole passes across a defect line, it changes its boundary type such that it now absorbs Pauli-$X$ string operators instead of Pauli-$Z$ string operators. Once the deformation procedure is completed and the hole is
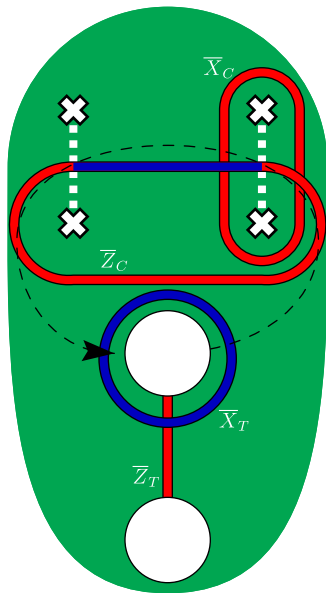


FIG. 11. The code-deformation scheme to entangle a qubit encoded with a pair of holes to a qubit encoded over four twists. At the top of the figure, we show the logical operators for a quadruple-twist qubit encoding, and at the bottom of the figure, we show the logical operators for a hole-pair qubit encoding. We present a scheme to perform a controlled-NOT gate where the qubit encoded with the four twists is the control qubit and the qubit encoded using holes is the target qubit. The black dashed line shows the trajectory that the hole in the middle of the figure must follow to complete a controlled-NOT gate between these qubits.
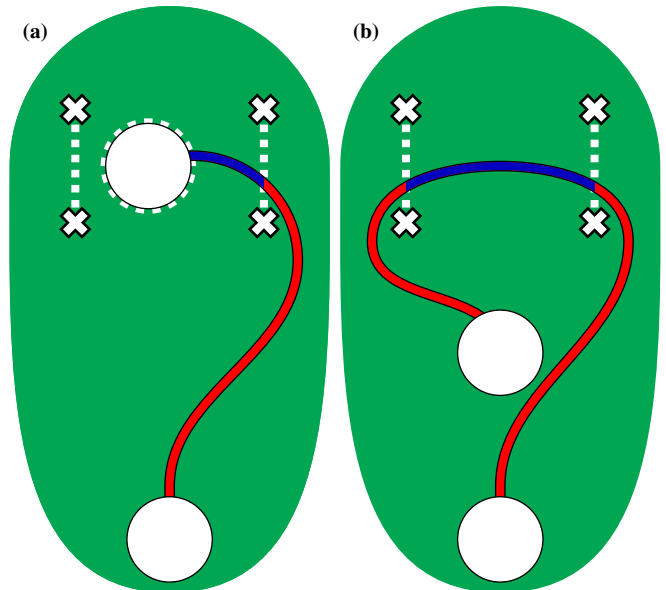


FIG. 12. The deformation of the $\bar{Z}_T$ operator under the fault-tolerant entangling operation. (a) The logical operator that terminates at the hole moves with the hole and is thus threaded between the twists. We also point out that the boundary of the hole changes from a rough boundary to a smooth boundary as it is passed across the defect line. As such, we draw a white dashed line around the puncture. (b) Upon moving the puncture back to its initial position, we see that we have wrapped the logical string operator around the lower two twists shown in the figure. One can check that the deformed logical operator is equivalent to $\bar{Z}_C \bar{Z}_T$, as shown in Fig. 11.

returned to its initial position, the logical operator is deformed around the lower two twists on the lattice, as we show in Ref. 12(b), which one can easily see is equivalent to $\bar{Z}_C\bar{Z}_T$ as depicted in Fig. 11.

We next consider the evolution of $\bar{X}_C$ as the hole is moved around the lattice. The operator $\bar{X}_C$ is a string of Pauli-$Z$ operators that wrap around the two twists shown on the right in Fig. 11. While the hole has a rough boundary, it can terminate Pauli-$Z$ strings and, as such, passes transparently through the logical operator. However, once the puncture passes over the defect line, the boundary changes from rough to smooth, and as such, the Pauli-$Z$ string can no longer terminate on the hole and must thus deform around the hole, as we see in Fig. 13(a). We continue to move the hole around its prescribed trajectory to its initial position, deforming the logical operator further, such that we finally obtain the logical operator shown in Fig. 13(b). It can easily be checked that this can be deformed by stabilizer manipulations to the logical operator $\bar{X}_C\bar{X}_T$ as drawn in Fig. 11.

Finally, it can readily be checked that the other two trivial relations, namely, $\bar{Z}_C \rightarrow \bar{Z}_C$ and $\bar{X}_T \rightarrow \bar{X}_T$, also hold under the code-deformation scheme. With this result, we show that the proposed braid completes a controlled-NOT gate between the two qubits shown in Fig. 11. We remark that, provided the holes maintain a width of $\mathcal{O}(d/4)$ and a separation of about $d$, and the twists are mutually separated
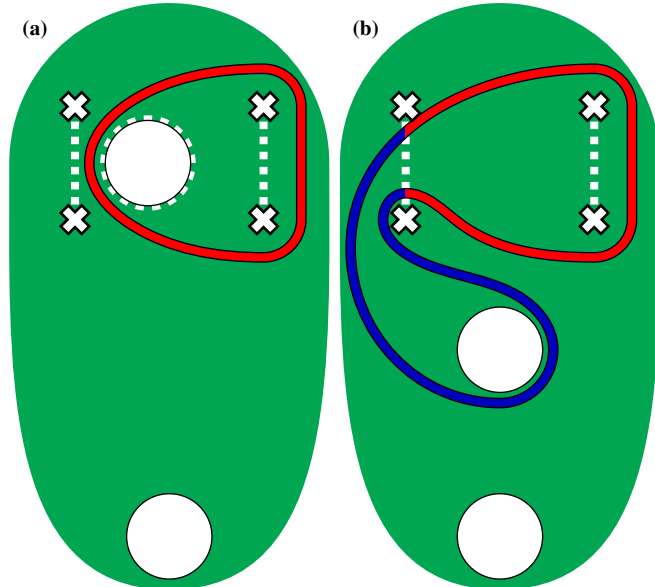


FIG. 13. The transformation $\bar{X}_C$ onto $\bar{X}_C\bar{X}_T$ under the code-deformation scheme. Diagram (a) shows how $\bar{X}_C$ is deformed around the hole when the hole lies in between the four twists. (b) Once the hole has been deformed around the two lower twists of the lattice and is returned to its initial position, the logical operator $\bar{X}_C$ is deformed onto a string operator that is equivalent up to stabilizer multiplication to the logical operator $\bar{X}_C\bar{X}_T$, as we show in Fig. 11.

by distance $d$, then this operation is completed without reducing the distance of the code below $d$. One can also show that we achieve a controlled-phase gate between the hole-pair qubit and the twist-quadruple qubit by deforming the hole around the two twists in the left side of the diagram. We leave this as an exercise for the reader.

Note that, once again, the effect of the braiding can be interpreted in terms of the anyons of $D(\mathbb{Z}_2)$. Given that a hole can support an electric excitation, and since two twists correspond to a single fermionic mode, which may fuse either to vacuum or to a $\psi$ excitation, the twists can be ignored and the braiding around the mode alone can be considered. The only instance in which a nontrivial braiding occurs is when both the hole and the fermionic mode are occupied by a nontrivial excitation. In this case, the $e$ anyon braids around the $\psi$ excitation, which yields a $-1$ phase, and hence we observe the application of a controlled operation.

As an aside, it may be interesting to reinterpret the proposed entangling operation by code deformation, followed by a logical measurement to teleport logical information between logical qubits as a gauge-fixing operation [28,48,49]. Specifically, this teleportation fault-tolerantly moves logical information between two different schemes of encoding. If we can regard the two logical qubits shown in Fig. 11 as one logical qubit and a second gauge qubit, then suitable preparation, or gauge fixing, of the gauge qubit, followed by the entangling operation that is achieved by code deformation, and subsequent measurement of the logical qubit will teleport information from the logical qubit to the gauge qubit. After the operation is completed, the qubit that originally supported the logical information is now a gauge qubit, and the gauge qubit maintains the logical information; as such, logical information has been switched between two different codes.

To understand this code-deformation operation as some special case of gauge fixing, we can interpret the system as the hole is braided as an intermediate code, or even a series of intermediate codes, where the nontrivial measurements that deform the code are elements of the gauge group of a subsystem code [89–91]. As before, the measurements we make to deform the hole can be regarded as a series of gauge-fixing operations. Finally, the measurement that is made to teleport logical information between the two codes after the entangling operation can be thought of as a third gauge-fixing operation that completes the transfer of logical information. It may be interesting to consider this example of code switching to help us shed light on the limitations and the potential applications of gauge fixing. We discuss this in more detail in Sec. VII B.

## B. Entangling twist qubits

Having shown that we can perform a controlled-NOT operation between a quadruple-twist encoded qubit and a hole-pair qubit, we can now use a hole-pair qubit as an ancilla qubit to perform a parity measurement between two

quadruple-twist qubits. Specifically, we take the ancilla qubit and perform entangling operations between both of the qubits involved in the parity measurement and subsequently measure the ancilla qubit.

As we have already seen, we entangle the ancilla qubit to a logical quadruple-twist qubit by braiding the hole around two of the twists that encode the qubit. It follows from this that, to perform a parity measurement between two logical qubits encoded with a quadruple of twists, we must braid a hole of an encoded ancillary hole pair around two of the twists of each qubit and then measure the hole-pair qubit by moving the two holes together. Given the ability to perform parity measurements, we can perform a controlled-NOT operation between qubits encoded with quadruples of twists by measurement given an additional ancillary twist-quadruple qubit. We describe this procedure below.

In Fig. 14, we show how to perform a controlled-NOT gate between twist qubits using parity measurements [4]. Having prepared the ancilla qubit in the $+1$ eigenstate of $\bar{X}_A$, we then measure $\bar{Z}_C\bar{Z}_A$, followed by $\bar{X}_A\bar{X}_T$, before measuring the ancilla qubit in the computational basis, thus projecting it onto an eigenstate of $\bar{Z}_A$.

Each of the three qubits in Fig. 14—the control, the ancilla, and the target qubit—are labeled in the center of each twist quadruple qubit. We use an encoding where the logical operator $\bar{Z}_\alpha(\bar{X}_\alpha)$ is a closed loop enclosing two vertically (horizontally) separated twists of each quadruple,
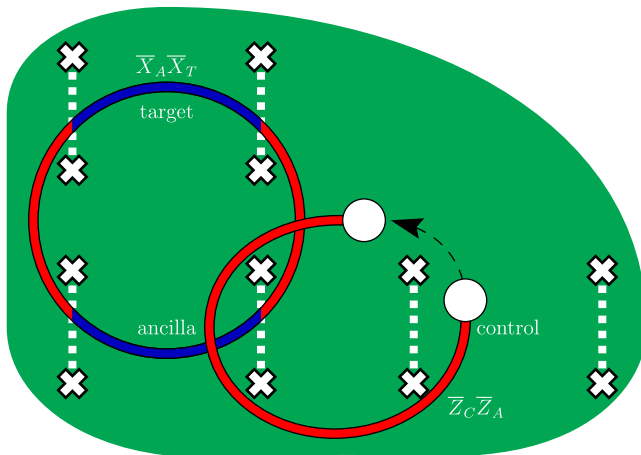


FIG. 14. Executing the two qubit parity measurements, we need to perform a controlled-NOT gate. Qubits are encoded with quadruples of twists, where the different logical qubits—the control, target, and ancilla—are labeled in the center of each twist quadruple. We measure $\bar{Z}_C\bar{Z}_A$ by braiding a single hole of a hole-pair qubit around two twists of the control qubit and two twists of the ancilla qubit, as shown in the figure. Upon completing this deformation, measuring the hole-pair qubit reveals the outcome of the parity measurement. We also show the $\bar{X}_A\bar{X}_T$ operator that we must measure. This is again achieved by braiding one hole of a hole pair, which is prepared in $\bar{X}$ around the loop followed by the $\bar{X}_A\bar{X}_T$ operator. Measuring $\bar{Z}_A$ completes the controlled-NOT gate, up to a Pauli correction.

as in Fig. 5(a) for $\alpha = C, A, T$. We also remark that with the chosen encoding, the total anyonic charge of a given quadruple qubit is vacuum. It follows from this result that the stabilizer group contains a closed loop of Pauli-$X$ and Pauli-$Z$ operators that encloses all four twists of each quadruple. A consequence of this is that it does not matter which pair of vertically (horizontally) aligned twists the $\bar{Z}_\alpha$ ($\bar{X}_\alpha$) enclose; both are equivalent up to multiplication by stabilizer group elements and thus have the same action on the code space.

Given these facts, together with the earlier discussion in this section where we show that we can entangle a hole-pair qubit to a quadruple-twist qubit, we can easily show how to perform a fault-tolerant controlled-NOT gate by parity measurements. First, to perform $\bar{Z}_C\bar{Z}_A$ we prepare a pair of holes in the $+1$ eigenvalue eigenstate of logical operator $\bar{X}_h$, where we have used a lowercase index to indicate that this is the logical qubit encoded by the hole pair. Next, knowing that braiding one of the two holes of the hole pair around two of the vertically aligned twists of a quadruple qubit results in a controlled-phase gate, it follows that braiding a hole around two of the vertically aligned twists of the ancilla qubit and then two of the vertically aligned twists of the control qubit and then returning the hole to its initial position will result in a controlled-phase gate between the hole-pair qubit and the ancilla qubit, and a controlled-phase gate between the hole qubit and the logical control qubit. In the bottom right-hand side of Fig. 14, we show how the $\bar{X}_h$ logical operator is deformed into a logical operator, which is equivalent up to stabilizers to the logical operator $\bar{Z}_C\bar{Z}_A\bar{X}_h$. Finally, following this entangling operation, measuring $\bar{X}_h$ returns the value of the fault-tolerant nondestructive parity measurement. This measurement is completed by moving the two holes of the hole pair back together, as we indicate by the black-dashed arrow in Fig. 14.

We must also perform a $\bar{X}_A\bar{X}_T$ gate to execute the fault-tolerant controlled-NOT gate. This high-weight logical operator is also shown in Fig. 14, where the logical operator is a string of Pauli operators that enclose two of the horizontally aligned twists of the ancilla qubit and two of the horizontally aligned qubits of the target qubit. To measure this logical operator fault-tolerantly, once again, we prepare an additional logical qubit using a pair of holes close to both the ancilla qubit and the target qubit in the $+1$ eigenstate of $\bar{X}_h$. We then deform one of the holes of the pair around two of the horizontally aligned qubits of both the ancilla qubit and the target qubit. This deformation effectively performs a controlled-NOT gate between the hole-pair qubit and the ancilla qubit, and a controlled-NOT gate between the hole-pair qubit and the target qubit. Once again, measuring the hole-pair qubit in the basis of eigenstates of the $\bar{X}_h$ operator completes the $\bar{X}_A\bar{X}_T$ parity measurement.

Finally, we perform the fault-tolerant $\bar{Z}_A$ measurement by producing a hole-pair qubit in an eigenstate of $\bar{X}_h$,

performing a controlled-phase gate as we have already described and then measuring $\bar{X}_h$. The outcome of this measurement determines the Pauli correction that we must apply to complete the measurement-only controlled-NOT gate. Together with the methods we have outlined in Sec. IV C, we can generate all of the gates of the Clifford group. Alongside noisy processes such as magic-state distillation [43], we can use the Clifford group to perform universal quantum computation with our scheme.

## VI. LATTICE SURGERY WITHIN THE TWIST FRAMEWORK

Finally, with the observation that the corners of the planar code can be regarded as Majorana modes, it is interesting to recognize that lattice surgery [29,30] is reminiscent of the measurement-only entangling gate scheme that we discuss in Sec. V B.

Lattice surgery provides a way of entangling pairs of planar codes within a two-dimensional architecture via fault-tolerant logical parity measurements. This entanglement scheme is particularly interesting from a practical perspective because, in principle, the planar codes of this computational architecture can be kept well separated while they are not interacting with other qubits. In this sense, the lattice-surgery computational scheme is modular. In contrast, braiding schemes, for instance, where all the qubits are encoded over twist defects or punctures, need to be kept on a common manifold, and as such, it will be necessary to design large surface-code architectures with these schemes, which can support all of the logical qubits that are needed to complete a computation.

In Fig. 15, we show how an entangling gate is performed between logical qubits encoded with planar codes via lattice surgery. The figure depicts three planar codes—an ancilla qubit, a control qubit, and a target qubit—together with the $\bar{Z}_C\bar{Z}_A$ operator and the $\bar{X}_A\bar{X}_T$ operator shown in red and blue, respectively, that are used to perform entangling gates via lattice surgery [29]. The figure also shows the additional qubits of the lattice used in the original measurement-only topological quantum-computation scheme that we have discussed above. The qubits used in the scheme above that are not required in lattice surgery are shown in pale colors. Instead, lattice surgery uses only $\mathcal{O}(L)$ qubits to perform two-qubit parity measurements. This is in contrast to the measurement-only scheme discussed above where $\mathcal{O}(L^2)$ qubits are required in between logical qubits encoded with twist quadruples on the lattice in order to maintain the distance of the code. Details on performing fault-tolerant logical parity measurements by lattice surgery are given in Refs. [29,30].

Following this observation, it may also be interesting to explore this picture further to discover new fault-tolerant schemes for quantum computation with low resource demands using other, more exotic topological models [22,92,93]. For instance, one might also consider reinterpreting lattice surgery with the color code from the point of
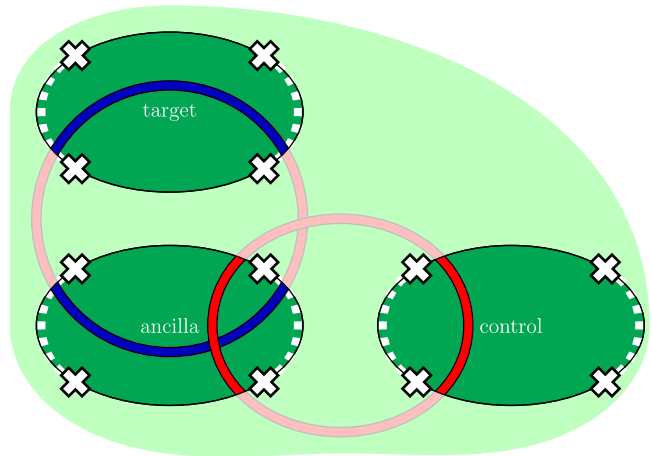


FIG. 15. Using the picture of twists, we can view lattice surgery as a measurement-only approach to performing logical gates between non-Abelian point particles. In bold colors, we show three planar codes, and the $\bar{Z}_C\bar{Z}_A$ and $\bar{X}_A\bar{X}_T$ measurements we need to perform lattice surgery are shown in red and blue, respectively. We also show with pale colors the qubits used in the measurement-only scheme above that are not required in the lattice surgery scheme. From this perspective, we see that lattice surgery is a measurement-only topological quantum computation scheme with a significant reduction in resource costs, as we are able to perform the required parity measurements with only $\mathcal{O}(L)$ ancillary physical qubits, which is in contrast to $\mathcal{O}(L^2)$ qubits if we maintain all of the twist defects on the same lattice.

view of twist defects. Lattice surgery and twist defects have been considered for computation with the color code in Refs. [30] and [21], respectively.

## VII. HYBRID ENCODING SCHEME

Having considered several different fault-tolerant schemes for encoding and manipulating quantum information, we finally introduce a new method of encoding logical qubits that makes use of both punctures and twist defects. We call qubits encoded in this fashion hybrid qubits. Entangling operations can be achieved without additional ancilla qubits, and we can perform one nontrivial single-qubit element of the Clifford group. With these encodings, we need only braid holes with different boundary types. More details on hybrid qubits are found in Ref. [42].

### A. Hybrid qubit

We show a hybrid qubit in Fig. 16, together with its logical operators. The figure shows two twist defects at the terminal points of a single defect line, together with one hole with a rough boundary and a second hole with a smooth boundary. The logical operator $\bar{Z}$ extends from one hole around a single twist defect and terminates at the other hole. The logical operator must follow this trajectory because the string must cross the defect line such that it can terminate at both of the punctures of the qubit. The
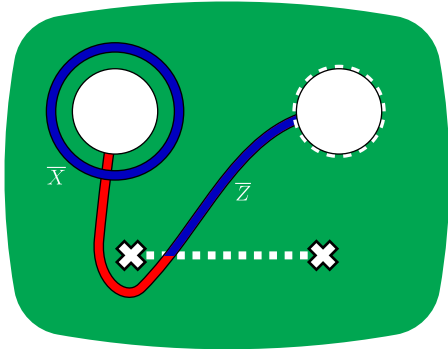
FIG. 16.    A hybrid qubit is encoded using a pair of holes—one with a rough boundary and one with a smooth boundary—and one pair of twist defects. The two anticommuting logical operators of the hybrid qubit are shown.

logical operator $\bar{X}$ is a string that forms a loop that encloses a single puncture of the hybrid qubit, as seen in the figure. To achieve code distance $d$, the two holes must each have a circumference $\mathcal{O}(d)$, the two twist defects must be separated from each other by a distance of about $d$, and both defects must be separated from the two holes by a distance $\mathcal{O}(d/2)$.

We entangle pairs of hybrid qubits by exchanging the smooth hole of one hybrid qubit with the rough hole of the second hybrid qubit. We can also perform the single-qubit Clifford operation $B_1$ by braiding the two holes of a hybrid qubit or by exchanging the two twist defects. It is interesting that we can combine punctures and dislocation lines to encode qubits. Such combinations could potentially be used to discover better encoding rates of logical qubits to physical qubits than those currently known. We leave such a calculation to future work. Similar work in this direction is given in Refs. [41,64]. However, finding the optimal rate for encoding qubits remains an open problem. In what follows, we describe how to fault-tolerantly convert between different encodings that may allow us to exploit the benefits of each qubit type.

### B. Switching between different encodings

We have now identified and discussed three distinct methods of encoding logical qubits in the surface code, namely, by hole pairs, twist quadruples, and with hybrid qubits, all of which have distinct capabilities of performing logical operations. Given their complementary properties, it is interesting to see that we can fault-tolerantly switch between these different encodings. We have already outlined how we can switch between a hole-pair qubit and a twist-quadruple qubit in Sec. V A. In what follows, we briefly describe how to switch between a hybrid encoding and the other two encodings, thus providing a direct path to switch between any two of the three encodings. We remark that the switching procedures we give can be understood naturally in the anyonic picture, as they are simply transferring the anyons from one occupational mode to another

without allowing them to be measured. We also point out that this idea of code switching is reminiscent of the ideas presented in Ref. [87], where code switching is used to complete a universal gate set with parity measurements for a particular non-Abelian anyon model.

We first consider transferring a logical qubit from a hybrid qubit to a twist qubit. To do so, we first prepare a second pair of twist defects some distance at least $\mathcal{O}(d/2)$ from the holes of the hybrid qubit, and a distance at least $\mathcal{O}(d)$ from the two twist defects of the hybrid qubit. Then, one of the two holes, say, the hole with a smooth boundary of the hybrid qubit, is braided around one of the two new twist defects by code deformation and then returned to its initial position. The braid operation moves the hole across a defect line and thus changes the boundary type of the hole.

After completing the braid, we can measure a string of Pauli-$Z$ operators that terminates at the boundaries of the two distinct holes, which maps the logical qubit onto a logical encoding of the four twists that now remain on the lattice, up to some Pauli correction. The Pauli correction we must apply is determined by the outcome of the string-operator measurement. The string-operator measurement can be performed fault-tolerantly by moving the two holes together to form a single hole. The single remaining hole can subsequently be closed, leaving a single logical qubit encoded over four twists on the lattice, thus completing the code-switching operation.

We can also map from a twist-quadruple qubit to a hybrid qubit. This is achieved by preparing a pair of holes on the lattice a distance $\mathcal{O}(d/2)$ away from the twist defects. Then, we braid one of the two holes around one of the twist defects of the twist-quadruple qubit and return the hole to its original position. Finally, we measure a loop operator that encloses two twists, including the one twist that was braided with the hole, to teleport the encoded logical information onto a hybrid qubit up to a Pauli correction, which is determined by the outcome of the loop measurement. Remaining on the lattice is a hybrid qubit that is made up of two holes on the lattice and two twist defects that were not enclosed by the loop-operator measurement. The other two twists that remain on the lattice can be removed by code deformation.

Finally, to map between a hybrid qubit and a hole-pair qubit, we simply braid the hole with a smooth boundary of the hybrid qubit around one of the two twist defects of the hybrid qubit, which transforms the boundary type of the braided hole. We are then free to remove the twists from the lattice as the logical information is now preserved in the hole-pair qubit. The reverse operation can be performed such that a hole-pair qubit is mapped onto a hybrid qubit by preparing a pair of twist defects on the lattice and then braiding one of the holes of the hole pair around one of the new twist defects. The two twist defects and the two holes now compose a hybrid qubit describing the logical information that was initially encoded by the hole-pair qubit.

Given that the three different encodings of logical qubits on the surface code all have complementary properties, it is interesting to find the most resource-efficient method of encoding and manipulating logical qubits. To remind the reader, the twist qubit can generate all of the single-qubit Clifford operations by braiding, but it requires a logical ancilla qubit to perform entangling gates using parity measurements, the double code of Sec. IV D being a notable exception. In contrast, hole-pair qubits can be entangled readily, but they do not achieve single-qubit Clifford gates by braiding. Hybrid qubits fall in the middle ground of these two examples, as they can be directly entangled without ancilla and can perform a subset of single-qubit Clifford rotations. Given that we are also capable of efficiently switching between these different types of qubits, it may be interesting to try to discover more efficient computational schemes in space-time resource costs using code switching. We also mention again that it may be an interesting direction of study to reinterpret the examples of code-deformation gauge-fixing schemes from a more fundamental point of view.

## VIII. CONCLUDING REMARKS

To summarize, we have unified several methods of manipulating logical qubits with the surface code. Notably, we have demonstrated new code-deformation schemes to implement the full Clifford group using a small number of weight-five local measurements over the planar-code lattice. In contrast, surface-code quantum computation using only hole defects or standard lattice surgery may be more resource intensive once the implementation of full algorithms is considered, as with these architectures we must prepare and maintain additional logical qubits in eigenstates of the Pauli-$Y$ matrix via a distillation scheme to complete the Clifford group. Alternatively, the two-dimensional color code, which achieves the full Clifford group transversally, requires weight-six stabilizer measurements, which we expect to be more challenging to perform in the laboratory. With these considerations, we argue that the new deformation procedures we have considered may lead to fault-tolerant quantum-computational schemes with lower resource costs than previously considered architectures. To investigate our corner-braiding scheme further, we should examine how it behaves at the circuit level under a realistic noise model as gates are performed. We leave such an analysis to future work.

We have also built on the analogy between the corners of the planar code, twists, and Ising anyons to show that lattice surgery fits into the more conventional picture of measurement-only topological quantum computation. We suggest that this observation may be extended to other topological phases with boundaries to develop other fault-tolerant quantum-computational schemes with lattice surgery. Certainly, it may be instructive to find modular quantum-computational models where a universal gate set is achieved between qubits encoded on some suitably chosen topological

substrates via fault-tolerant logical parity measurements. It might also be interesting to adapt the schemes we develop here for use in a fault-tolerant measurement-based scheme [11,12,17,94] for quantum computation. Such an extension may make some of the present ideas experimentally amenable to a linear optical architecture [95].

―――――

[1] A. Y. Kitaev, *Fault-Tolerant Quantum Computation by Anyons*, Ann. Phys. (Amsterdam) **303**, 2 (2003).

[2] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, *Topological Quantum Memory*, J. Math. Phys. (N.Y.) **43**, 4452 (2002).

[3] *Quantum Error Correction*, edited by D. A. Lidar and T. A. Brun (publisher Cambridge University Press, Cambridge, England, 2013).

[4] B. M. Terhal, *Quantum Error Correction for Quantum Memories*, Rev. Mod. Phys. **87**, 307 (2015).

[5] B. J. Brown, D. Loss, J. K. Pachos, C. N. Self, and J. R. Wootton, *Quantum Memories at Finite Temperature*, Rev. Mod. Phys. **88**, 045005 (2016).

[6] J. Preskill, Lecture Notes on Quantum Computation, 2004, http://www.theory.caltech.edu/people/preskill/ph229/#lecture.

[7] C. Nayak, S. H. Simon, A. Stern, M. Freedman, and S. D. Sarma, *Non-Abelian Anyons and Topological Quantum Computation*, Rev. Mod. Phys. **80**, 1083 (2008).

[8] J. K. Pachos, *Introduction to Topological Quantum Computation* (Cambridge University Press, Cambridge, England, 2012).

[9] A. Kitaev, *Anyons in an Exactly Solved Model and Beyond*, Ann. Phys. (Amsterdam) **321**, 2 (2006).

[10] R. Raussendorf, J. Harrington, and K. Goyal, *A Fault-Tolerant One-Way Quantum Computer*, Ann. Phys. (Amsterdam) **321**, 2242 (2006).

[11] R. Raussendorf and J. Harrington, *Fault-Tolerant Quantum Computation with High Threshold in Two Dimensions*, Phys. Rev. Lett. **98**, 190504 (2007).

[12] R. Raussendorf, J. Harrington, and K. Goyal, *Topological Fault-Tolerance in Cluster State Quantum Computation*, New J. Phys. **9**, 199 (2007).

[13] H. Bombin and M. A. Martin-Delgado, *Interferometry-Free Protocol for Demonstrating Topological Order*, Phys. Rev. B **78**, 165128 (2008).

[14] H. Bombin and M. A. Martin-Delgado, *Quantum Measurements and Gates by Code Deformation*, J. Phys. A **42**, 095302 (2009).

[15] A. G. Fowler, A. M. Stephens, and P. Groszkowski, *High-Threshold Universal Quantum Computation on the Surface Code*, Phys. Rev. A **80**, 052312 (2009).

[16] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, *Surface Codes: Towards Practical Large-Scale Quantum Computation*, Phys. Rev. A **86**, 032324 (2012).

[17] C. G. Brell, *Generalized Cluster States Based on Finite Groups*, New J. Phys. **17**, 023029 (2015).

[18] I. Cong, M. Cheng, and Z. Wang, *Topological Quantum Computation with Gapped Boundaries*, arXiv:1609.02037.

[19] I. Cong, M. Cheng, and Z. Wang, *On Defects between Gapped Boundaries in Two-Dimensional Topological Phases of Matter*, arXiv:1703.03564 [Phys. Rev. X (to be published)].

[20] H. Bombin, *Topological Order with a Twist: Ising Anyons from an Abelian Model*, Phys. Rev. Lett. **105**, 030403 (2010).

[21] H. Bombin, *Clifford Gates by Code Deformation*, New J. Phys. **13**, 043005 (2011).

[22] M. Barkeshli, C.-M. Jian, and X.-L. Qi, *Twist Defects and Projective Non-Abelian Braiding Statistics*, Phys. Rev. B **87**, 045130 (2013).

[23] M. Barkeshli, P. Bonderson, M. Cheng, and Z. Wang, *Symmetry, Defects, and Gauging of Topological Phases*, arXiv:1410.4540 [Phys. Rev. X (to be published)].

[24] M. B. Hastings and A. Geller, *Reduced Space-Time and Time Costs Ising Dislocation Codes and Arbitrary Ancillas*, Quantum Inf. Comput. **15**, 0962 (2015).

[25] J. R. Wootton, *A Family of Stabilizer Codes for $d(\mathbb{Z}_2)$ Anyons and Majorana Modes*, J. Phys. A **48**, 215302 (2015).

[26] H. Bombin and M. A. Martin-Delagado, *Topological Quantum Distillation*, Phys. Rev. Lett. **97**, 180501 (2006).

[27] H. Bombin and M. A. Martin-Delgado, *Topological Computation without Braiding*, Phys. Rev. Lett. **98**, 160502 (2007).

[28] H. Bombín, *Gauge Color Codes: Optimal Transversal Gates and Gauge Fixing in Topological Stabilizer Codes*, New J. Phys. **17**, 083002 (2015).

[29] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, *Surface Code Quantum Computing by Lattice Surgery*, New J. Phys. **14**, 123011 (2012).

[30] A. J. Landahl and C. Ryan-Anderson, *Quantum Computing by Color-Code Lattice Surgery*, arXiv:1407.5103.

[31] S. B. Bravyi and A. Y. Kitaev, *Quantum Codes on a Lattice with Boundary*, arXiv:quant-ph/9811052.

[32] S. Beigi, P. W. Shor, and D. Whalen, *The Quantum Double Model with Boundary: Condensations and Symmetries*, Commun. Math. Phys. **306**, 663 (2011).

[33] A. Kitaev and L. Kong, *Models for Gapped Boundaries and Domain Walls*, Commun. Math. Phys. **313**, 351 (2012).

[34] N. H. Lindner, E. Berg, G. Refael, and A. Stern, *Fractionalizing Majorana Fermions: Non-Abelian Statistics on the Edges of Abelian Quantum Hall States*, Phys. Rev. X **2**, 041002 (2012).

[35] M. Levin, *Protected Edge Modes without Symmetry*, Phys. Rev. X **3**, 021009 (2013).

[36] M. Barkeshli, C.-M. Jian, and X.-L. Qi, *Theory of Defects in Abelian Topological States*, Phys. Rev. B **88**, 235103 (2013).

[37] B. J. Brown, S. D. Bartlett, A. C. Doherty, and S. D. Barrett, *Topological Entanglement Entropy with a Twist*, Phys. Rev. Lett. **111**, 220402 (2013).

[38] H. Zheng, A. Dua, and L. Jiang, *Demonstrating Non-Abelian Statistics of Majorana Fermions Using Twist Defects*, Phys. Rev. B **92**, 245139 (2015).

[39] P. Bonderson, M. Freedman, and C. Nayak, *Measurement-Only Topological Quantum Computation*, Phys. Rev. Lett. **101**, 010501 (2008).

[40] P. Bonderson, M. Freedman, and C. Nayak, *Measurement-Only Topological Quantum Computation via Anyonic Interferometry*, Ann. Phys. (Amsterdam) **324**, 787 (2009).

[41] T. J. Yoder and I. H. Kim, *The Surface Code with a Twist*, Quantum **1**, 2 (2017).

[42] K. Laubscher and M. Kesselring, Master's project work (2016), https://drive.google.com/open?id=0B2onG4gv2WeEMkxlRWhlcjlPQU0.

[43] S. Bravyi and A. Kitaev, *Universal Quantum Computation with Ideal Clifford Gates and Noisy Ancillas*, Phys. Rev. A **71**, 022316 (2005).

[44] E. T. Campbell, B. M. Terhal, and C. Vuillot, *The Steep Road Towards Robust and Universal Quantum Computation*, arXiv:1612.07330.

[45] A. G. Fowler and S. J. Devitt, *A Bridge to Lower Overhead Quantum Computation*, arXiv:1209.0510 [Phys. Rev. Lett. (to be published)].

[46] P. Aliferis, Ph.D. thesis, School California Institute of Technology (2007).

[47] N. Cody Jones, R. Van Meter, A. G. Fowler, P. L. McMahon, J. Kim, T. D. Ladd, and Y. Yamamoto, *Layered Architecture for Quantum Computing*, Phys. Rev. X **2**, 031007 (2012).

[48] A. Paetznick and B. W. Reichardt, *Universal Fault-Tolerant Quantum Computation with Only Transversal Gates and Error Correction*, Phys. Rev. Lett. **111**, 090505 (2013).

[49] J. T. Anderson, G. Duclos-Cianci, and D. Poulin, *Fault-Tolerant Conversion between the Steane and Reed-Muller Quantum Codes*, Phys. Rev. Lett. **113**, 080501 (2014).

[50] H. Bombín, *Dimensional Jump in Quantum Error Correction*, New J. Phys. **18**, 043038 (2016).

[51] S. Bravyi and A. Cross, *Doubled Color Codes*, arXiv:1509.03239.

[52] T. Jochym-O'Connor and S. D. Bartlett, *Stacked Codes: Universal Fault-Tolerant Quantum Computation in a Two-Dimensional Layout*, Phys. Rev. A **93**, 022323 (2016).

[53] C. Jones, P. Brooks, and J. Harrington, *Gauge Color Codes in Two Dimensions*, Phys. Rev. A **93**, 052332 (2016).

[54] B. J. Brown, N. H. Nickerson, and D. E. Browne, *Fault-Tolerant Error Correction with the Gauge Color Code*, Nat. Commun. **7**, 12302 (2016).

[55] E. T. Campbell and M. Howard, *Unifying Gate-Synthesis and Magic State Distillation*, Phys. Rev. Lett. **118**, 060501 (2017).

[56] E. T. Campbell and M. Howard, *A Unified Framework for Magic State Distillation and Multi-Qubit Gate-Synthesis*

*with Reduced Resource Cost*, Phys. Rev. A **95,** 022316 (2017).

[57] H. Bombín, *Single-Shot Fault-Tolerant Quantum Error Correction*, Phys. Rev. X **5,** 031043 (2015).

[58] D. Gottesman, Ph.D. thesis, California Institute of Technology (2001).

[59] S. Bravyi and B. Terhal, *A No-Go Theorem for a Two-Dimensional Self-Correction Quantum Memory Based on Stabilizer Codes*, New J. Phys. **11,** 043029 (2009).

[60] G. Moore and N. Read, *Nonabelions in the Fractional Quantum Hall Effect*, Nucl. Phys. **B360,** 362 (1991).

[61] X.-G. Wen, *Quantum Orders in an Exact Soluble Model*, Phys. Rev. Lett. **90,** 016803 (2003).

[62] Z. Nussinov and G. Ortiz, *A Symmetry Principle for Topological Quantum Order*, Ann. Phys. (Amsterdam) **324,** 977 (2009).

[63] B. J. Brown, W. Son, C. V. Kraus, R. Fazio, and V. Vedral, *Generating Topological Order from a Two-Dimensional Cluster State*, New J. Phys. **13,** 065010 (2011).

[64] N. Delfosse, P. Iyer, and D. Poulin, *Generalized Surface Codes and Packing of Logical Qubits*, arXiv:1606.07116.

[65] H. Zheng, A. Dua, and L. Jiang, *Measurement-Only Topological Quantum Computation without Forced Measurements*, New J. Phys. **18,** 123027 (2016).

[66] J. C. Y. Teo, *Globally Symmetric Topological Phase: From Anyonic Symmetry to Twist Defect*, J. Phys. Condens. Matter **28,** 143001 (2016).

[67] J. C. Y. Teo, A. Roy, and X. Chen, *Unconventional Fusion and Braiding of Topological Defects in a Lattice Model*, Phys. Rev. B **90,** 115118 (2014).

[68] Precisely, given an operator $O = O_A \otimes O_B$, the restriction of $O$ onto $A$ is $O_A \otimes \mathbb{1}_B$.

[69] S. Bravyi, *Universal Quantum Computation with the $\nu = 5/2$ Fractional Quantum Hall State*, Phys. Rev. A **73,** 042313 (2006).

[70] Y.-Z. You and X.-G. Wen, *Projective Non-Abelian Statistics of Dislocation Defects in a $\mathbb{Z}_n$ Rotor Model*, Phys. Rev. B **86,** 161107(R) (2012).

[71] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2000).

[72] D. Bacon and S. T. Flammia, *Adiabatic Cluster-State Quantum Computing*, Phys. Rev. A **82,** 030303(R) (2010).

[73] C. Cesare, A. J. Landahl, D. Bacon, S. T. Flammia, and A. Neels, *Adiabatic Topological Quantum Computing*, Phys. Rev. A **92,** 012336 (2015).

[74] S. Bravyi, *Subsystem Codes with Spatially Local Generators*, Phys. Rev. A **83,** 012320 (2011).

[75] Y.-Z. You, C.-M. Jian, and X.-G. Wen, *Synthetic Non-Abelian Statistics by Abelian Anyon Condensation*, Phys. Rev. B **87,** 045106 (2013).

[76] A. G. Fowler, *Low-Overhead Surface Code Logical Hadamard*, Quantum Inf. Comput. **12,** 0970 (2012).

[77] S. B. Bravyi and A. Y. Kitaev, *Fermionic Quantum Computation*, Ann. Phys. (Amsterdam) **298,** 210 (2002).

[78] J. Chiaverini, D. Leibfried, T. Schaetz, M. D. Barrett, R. B. Blakestad, J. Britton, W. M. Itano, J. D. Jost, E. Knilll, C. Langer, R. Ozeri, and D. J. Wineland, *Realization of Quantum Error Correction*, Nature (London) **432,** 602 (2004).

[79] M. D. Reed, L. DiCarlo, S. E. Nigg, L. Sun, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf, *Realization of Three-Qubit Quantum Error Correction with Superconducting Circuits*, Nature (London) **482,** 382 (2012).

[80] D. Nigg, M. Müller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt, *Quantum Computations on a Topologically Encoded Qubit*, Science **345,** 302 (2014).

[81] A. D. Córcoles, E. Magesan, S. J. Srinivasan, A. W. Cross, M. Steffen, J. M. Gambetta, and J. M. Chow, *Demonstration of a Quantum Error Detection Code Using a Square Lattice of Four Superconducing Qubits*, Nat. Commun. **6,** 6979 (2015).

[82] J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Yu Chen *et al.*, *State Preservation by Repetitive Error Detection in a Superconducting Quantum Circuit*, Nature (London) **519,** 66 (2015).

[83] M. Takita, A. D. Córcoles, E. Megesan, B. Abdo, M. Brink, A. Cross, J. M. Chow, and J. M. Gambetta, *Demonstration of Weight-for Parity Measurements in the Surface Code Architecture*, Phys. Rev. Lett. **117,** 210505 (2016).

[84] N. Ofek, A. Petrenko, R. Heeres, P. Reinhold, Z. Leghtas, B. Vlastakis, Y. Lio, L. Frunzio, S. M. Girvin *et al.*, *Demonstrating Quantum Error Correction that Extends the Lifetime of Quantum Information*, Nature (London) **536,** 441 (2016).

[85] S. M. Albrecht, A. P. Higginbotham, M. Madsen, F. Kuemmeth, T. S. Jespersen, J. Nygå rd, P. Krogstrup, and C. M. Marcus, *Exponential Protection of Zero Modes in Majorana Islands*, Nature (London) **531,** 206 (2016).

[86] O. Zilberberg, B. Braunecker, and D. Loss, *Controlled-NOT Gate for Multiparticle Qubits and Topological Quantum Computation Based on Parity Measurements*, Phys. Rev. A **77,** 012327 (2008).

[87] C. Levaillant, B. Bauer, M. Freedman, Z. Wang, and P. Bonderson, *Universal Gates via Fusion and Measurement Operations on $su(2)_4$ Anyons*, Phys. Rev. A **92,** 012301 (2015).

[88] M. Freedman and C. Levaillant, *Interferometric versus Projective Measurement of Anyons*, arXiv:1501.01339.

[89] D. Kribs, R. Laflamme, and D. Poulin, *Unified and Generalized Approach to Quantum Error Correction*, Phys. Rev. Lett. **94,** 180501 (2005).

[90] D. Poulin, *Stabilizer Formalism for Operator Quantum Error Correction*, Phys. Rev. Lett. **95,** 230504 (2005).

[91] D. W. Kribs, R. Laflamme, D. Poulin, and M. Lesosky, *Operator Quantum Error Correction*, Quantum Inf. Comput. **6,** 382 (2006).

[92] M. A. Levin and X.-G. Wen, *String-Net Condensation: A Physical Mechanism for Topological Phases*, Phys. Rev. B **71,** 045110 (2005).

[93] R. Koenig, G. Kuperberg, and B. W. Reichardt, *Quantum Computation with Turaev-Viro Codes*, Ann. Phys. (Amsterdam) **325,** 2707 (2010).

[94] A. Bolt, G. Duclos-Cianci, D. Poulin, and T. M. Stace, *Foliated Quantum Error-Correcting Codes*, Phys. Rev. Lett. **117,** 070501 (2016).

[95] T. Rudolph, *Why I Am Optimisitc about the Silicon-Photonic Route to Quantum Computing*, arXiv:1607.08535.