Systematic Algebraic Method to Identify Clifford Operations for Quantum Error-Correction Codes

Sheng-Chen Liu[1], Yu-Xuan Lin[1], Ying-Xiang Wang[1], and Liang-You Peng[1,2*]

[1]*State Key Laboratory for Mesoscopic Physics and Frontiers Science Center for Nano-optoelectronics, School of Physics, Peking University, Beijing 100871, China*
[2]*Collaborative Innovation Center of Extreme Optics, Shanxi University, Taiyuan 030006, China*

(Received 12 December 2025; accepted manuscript online 10 March 2026)

In the domain of quantum error correction, a critical task involves identifying logical operations on logical qubits for various quantum codes. However, owing to the inherent complexity of many quantum codes, devising an efficient method to implement the desired logical operations utilizing the structure of these codes presents a significant challenge. In previous studies, several methods were used to realize specific logical operations for certain quantum codes; however, they usually do not work for other quantum codes. In this study, we propose a systematic method to find solutions to any Clifford logical operation for Calderbank–Shor–Steane codes, which represent a broad category of many quantum codes. In contrast to previous geometrical methods, our method is based on an algebraic approach. The solved logical operation can be further decomposed into a sequence of single-qubit and two-qubit gates that can be directly realized using quantum hardware. By applying our method, we can find many geometrically difficult logical operations for two different types of quantum code, that is, toric and hyperbolic surface codes. We believe that the present study will find wide applications in the design of logical operations for many quantum low-density parity-check codes, which can play an important role in future fault-tolerant quantum computing.

DOI: 10.1088/0256-307X/43/4/040603 CSTR: 32039.14.0256-307X.43.4.040603

Introduction. Quantum error correction (QEC) is progressively recognized as a crucial strategy for advancing future quantum computing technologies, which can solve practical problems in different fields, such as cryptography,[1] optimization.[2] In recent decades, an increasing number of QEC code examples have been successfully demonstrated on various quantum computation platforms. The seven-qubit Steane code,[3] which is the smallest realization of color code,[4] is commonly used in experiments based on trapped-ion systems.[5–7] For improved scalability, the surface code[8] implemented on superconducting qubits is a generally accepted method.[9,10] Specifically, the code distance can be conveniently increased, and logical operations can be provided by lattice surgery.[11] Recently, neutral-atom arrays have developed rapidly and can support several hundred controllable physical qubits,[12,13] providing a new method for large-scale QEC codes.

However, practical quantum computations require numerous logical qubits and convenient fault-tolerant logical operations. For surface codes, the conventional method of creating logical qubits by cutting holes may waste many physical qubits, whereas logical operations are based on complicated movements of stabilizers. Therefore, we attempted to identify QEC codes that are more efficient than surface codes. Therefore, quantum low-density parity-check (LDPC) codes[14–16] have gradually attracted attention. LDPC codes are of many different types that are designed in several mathematical ways, such as homological algebra and the geometry of manifolds.[17–20] The common goal is to increase the encoding rates under the same conditions. While quantum information is encoded in a more compact manner, the connection between physical qubits becomes more complicated, which is a significant challenge for real quantum hardware. To solve this problem, a trapped-ion system with an all-to-all connectivity[21] is a promising candidate. For systems with limited connectivity, such as superconducting systems, we admit that without a special design, a number of additional SWAP gates proportional to the code distance and the original two-qubit gates in a quantum circuit are needed. However, many attempts have been made to make the connectivity of physical qubits compatible with the structure of some LDPC codes to avoid excessive physical resources.[22,23]

In quantum LDPC codes research, designing new codes is challenging, while finding logical operations of logical qubits is arduous. For geometry-based codes, researchers seek geometrical methods for logical operations. The hyperbolic surface code[24,25] enables CNOT gates between logical qubits through Dehn twists.[26,27] However, other logical gates like Hadamard or phase gates are difficult to realize, with no method to find such operations for every hyperbolic surface code. To address this limitation,

*Corresponding author. Email: liangyou.peng@pku.edu.cn
© 2026 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

we propose a systematic method. Because logical qubit information scatters across the physical qubits' surface, applying logical gates geometrically is challenging. We adopt an algebraic approach instead. By focusing on the target logical operation's effect while maintaining code structure through stabilizers, we can formulate algebra equations. While general representation space may be exponential with physical qubits, Clifford operations can be represented with linearly increasing dimension. These operations include CNOT, Hadamard, and phase gates. We develop a systematic method to construct and solve these equations, calculating physical gate sequences to implement Clifford logical operations for any Calderbank–Shor–Steane (CSS) code. In summary, this study provides a novel method for calculating the circuit realization of logical operations for a large number of QEC codes.

Method for Finding Logical Clifford Operation. We discuss the main idea of our algebraic method for solving various logical Clifford operations for CSS codes whose stabilizer groups can be generated by chains of either Pauli Z operators or Pauli X operators.[28] Note that most LDPC codes are CSS codes, whereas other non-CSS codes can also be mapped to CSS codes.[29] In the following, we concentrate on CSS codes without loss of generality but note that code mapping will lead to more physical qubits.

For a specific CSS code with k logical qubits encoded by n physical qubits, n_Z independent Z stabilizers and n_X independent X stabilizers exist. Each stabilizer can be represented by a row vector belonging to $\mathbb{F}_2^n$, where the locations of 1 denote qubits operated by the corresponding Pauli Z or Pauli X. Then, we collect these stabilizers to form the parity-check matrix H_Z (H_X) with dimensions $n_Z \times n$ ($n_X \times n$). Because all the stabilizers commute with each other, we have

$$H_Z H_X^{\mathrm{T}} = 0. \tag{1}$$

We emphasize that such matrix calculations are performed in $\mathbb{F}_2$ unless otherwise specified. Furthermore, any Pauli chain on n qubits can be represented by the row vector $A_{\mathrm{P}} = (A_Z|A_X)$ in $\mathbb{F}_2^{2n}$, where A_Z and A_X are row vectors in $\mathbb{F}_2^n$. For the ith qubit, the pair $\big((A_Z)_i, (A_X)_i\big)$ admits four distinct cases: $(0,0) \to I_i$, $(1,0) \to Z_i$, $(0,1) \to X_i$, and $(1,1) \to Y_i$.

Next, we examine the Clifford operations in the n qubits system. In the Heisenberg picture, any Clifford operation maps one Pauli chain to another. If we only consider the operations that do not produce the minus sign of the Pauli chains with only Z or only X (the resultant Pauli chains can have any type of Pauli operators), then each operation can be regarded as a linear mapping on $\mathbb{F}_2^{2n}$ and represented by a $2n \times 2n$ binary matrix U.[30] After the operation, the Pauli chain becomes $A_{\mathrm{P}}U$. The sign problem can be easily solved using Pauli Z and X; thus, we mainly discuss the operations that are restricted in the above-mentioned way. Not all binary matrices U are capable of representing a Clifford operation. Leveraging two key properties, namely, any two Pauli chains either commute or anti-commute and Clifford operations preserve this commutation relationship, we can conclude that U must be a symplectic matrix:[31]

$$U\Lambda U^{\mathrm{T}} = \Lambda, \tag{2}$$

where $\Lambda = \begin{pmatrix} 0 & I_n \\ I_n & 0 \end{pmatrix}$. Therefore, the representation for Pauli chains and Clifford operations in binary space is called symplectic representation hereafter.

We must select the proper symplectic matrices to realize the target logical Clifford operation for a certain code. First, the code structure is preserved. Under the transformation of U, the stabilizers cannot change to Pauli chains that are not in the original stabilizer group but can permute with each other or recombine after the operation, such as some methods of logical Hadamard gates on the toric code.[32] For simplicity, we further restrict the stabilizers to keep themselves after U, which yields

$$\begin{pmatrix} H_Z & 0 \\ 0 & H_X \end{pmatrix} U = \begin{pmatrix} H_Z & 0 \\ 0 & H_X \end{pmatrix}. \tag{3}$$

The transformation should also map all logical Pauli Z and Pauli X operators to other logical Pauli operators according to the specific Clifford operation. For CSS code, the logical Z (X) operators can be chosen as physical Pauli Z (X) chains; therefore, the mapping of logical Pauli operators can be written as

$$\begin{pmatrix} \bar{Z} & 0 \\ 0 & \bar{X} \end{pmatrix} U = \begin{pmatrix} \bar{Z}^Z & \bar{X}^Z \\ \bar{Z}^X & \bar{X}^X \end{pmatrix}, \tag{4}$$

where each block denotes a matrix of dimensions $k \times n$. Specifically, $\bar{Z}$ and $\bar{X}$ denote logical Pauli Z and Pauli X for k logical qubits, whereas the blocks on the right-hand side can be linearly expressed in terms of these.

Once the CSS code and target logical Clifford operation are specified, our objective is to solve the matrix U that satisfies Eqs. (2)–(4). We propose a comprehensive algorithm to obtain the solution, with the key challenge being the quadratic constraints in Eq. (2). The main idea is to regard some quadratic terms of the elements in U as new variables; then, the constraints will degrade into large linear equations. We note that the coefficient matrix of the linear equations, with row and column dimensions both on the order of n^2, is highly sparse, leading to a computational complexity of slightly over $O(n^4)$ rather than exponential. In the solution space, we carefully match the quadratic variables and elements in U. This step does not consume much computational resources. The algorithm details are presented in the Supplementary Materials (SM).

Physical Gate Decomposition. After solving the matrix U, the subsequent task is to decompose it into single-qubit and two-qubit physical gates, which can be conveniently realized in different platforms of quantum computation. In this section, we first introduce the relationships between several gates and matrix operations, which are closely related to Gaussian elimination. Next, we explain the process of decomposing U. Drawing on previous studies,[31,33,34] our method yields a clear algorithm suitable for deriving quantum circuits for logical Hadamard $\bar{H}$ and phase $\bar{S}$ operations.

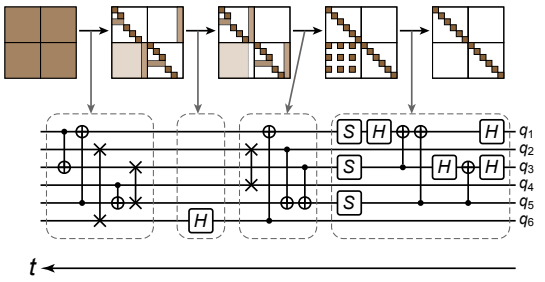

Fig. 1. Schematic of transformation matrix U decomposition. Four types of symplectic operations, which are related to four types of gates, can be used to eliminate U to I_{2n}. After the elimination, the sequence of gates is obtained for the realization of U. The process has four main steps, as the dashed boxes show. According to Eq. (6), the gates are applied in reverse order to realize U; therefore, the timeline points to the left.

Four types of physical gates are used. Here, we introduce the relationship between each gate and the corresponding operation on U.

1. Hadamard gate: H_i swaps the ith and $(i+n)$th columns.
2. Phase gate: S_i adds the $(i+n)$th column to the ith column.
3. CNOT gate: CNOT_{ij} adds the jth column to the ith column, in addition to adding the $(i+n)$th column to the $(j+n)$th column simultaneously.
4. SWAP gate: SWAP_{ij} swaps the ith and jth columns while swapping the $(i+n)$th and $(j+n)$th columns.

We provide a detailed description in the SM.

The next step is to decompose U by using these gates. We identified four types of elementary matrices, which do not include all elementary operations for $2n\times 2n$ matrices. However, they are symplectic operations that satisfy the condition of Eq. (2). Therefore, we focused on reducing U to I_{2n} using a sequence of elementary matrices:

$$UP_1P_2\cdots P_{T-1}P_T = I_{2n}. \tag{5}$$

After checking each type of P that satisfies $P^2 = I_{2n}$, we obtain

$$U = P_TP_{T-1}\cdots P_2P_1, \tag{6}$$

Subsequently, the sequence of gates for operation U can be restored according to the relationship between the elementary matrices and gates.

As discussed above, Gaussian elimination cannot be freely performed to obtain Eq. (5). Instead, we can use only four types of symplectic operation. Therefore, the decomposition process must be designed carefully. The typical process is illustrated in Fig. 1.

For two-qubit gates, focusing on the left $2n\times n$ part of elementary matrices shows all Gaussian elimination operations in $\mathbb{F}_2$ are contained. The first step converts the left part of U to reduced column echelon form, while the elementary matrices dually operate the right n columns, reducing the right $2n\times n$ part.

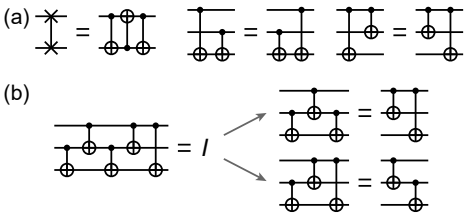

Fig. 2. Several equivalence relations of two-qubit gates. (a) One SWAP gate can be substituted by three CNOT gates. Two CNOT gates commute in some cases. (b) Template of five CNOT gates used to reduce the number of gates. According to the template, we can cut three gates to two in various ways.

The target of the Gaussian elimination is I_{2n}, so we hope that the $2n\times n$ reduced column echelon form already has I_n block on the top. In fact, the symplectic matrix U must be full rank, which means that after the first step, there is still no zero column. If we have not obtained I_n as stated above, some columns on the right will have zero elements on the top half, which cannot be further operated by the two-qubit gates. Now H gates can be used to swap these columns with the right part of U, then we focus the left part again with the operations of the two-qubit gates. Usually, the upper left and the lower right $n\times n$ blocks becomes I_n at this time.

However, some non-zero elements may still exist on the lower left or upper right blocks of U. For instance, we consider that there are only non-zero elements on the lower left block. For the element with index $(i+n,i)$, we can use S gate on the ith qubit to eliminate it. For the pair of elements with indexes $(i+n,j)$ and $(j+n,i)$ where $i<j$, we can use $H_i\mathrm{CNOT}_{ji}H_i$ to clean up them. If there are non-zero elements on the upper right block, some extra H gates can help the process. After this step, the matrix U is totally eliminated to I_{2n}, while our decomposition of Eq. (6) has been finished.

In actual examples of QEC codes, logical operations contain numerous two-qubit gates in the first and third steps. The circuit may not be the best realization of U. A SWAP gate equals three CNOT gates; thus, the two-qubit gates can be regarded as CNOT gates. A five-gate template[34,35] can reduce some CNOT gates. As shown in Fig. 2, the template reduces combinations of three CNOT gates to two gates. We search these combinations in both circuit sections. After substitution, new combinations may emerge, requiring repeated searches until no further simplification is possible. This yields the final decomposition circuit.

Results for LDPC Codes. We demonstrated the capability of our proposed method to realize Clifford operations for quantum LDPC codes. Two representative classes of LDPC codes, i.e., toric code[36,37] and hyperbolic surface code,[24,25] are investigated, with the aim of finding several Clifford operations that have not been addressed in prior studies. The results are presented in two ways. On the one hand, the decomposition algorithm can tell us the physical resources for the logical operations. On the other hand, the decomposed gates together with the stabilizers can be input to Stim,[38]

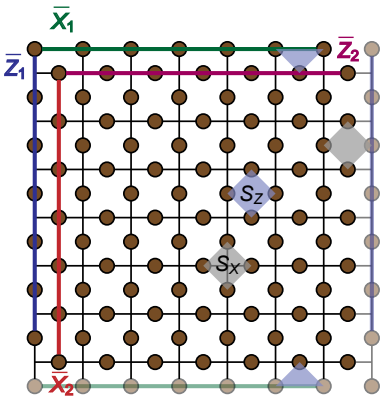

Fig. 3. Implementation of the toric code with $d = 7$. Physical qubits are represented by dark circles, and logical Pauli operators are denoted by colored lines. Only the dark circles denote the actual physical qubits, and the light circles are virtual physical qubits due to the boundary conditions. The stabilizers S_Z, S_X are defined by plaquettes and vertices of the array.

which is a QEC simulator, for the predicted performances of the logical operations. Although the entire algorithm has generality for different codes and logical operations, it must sacrifice some fault tolerance. Traditionally, each logical operation is specifically designed for each QEC code; however, this cannot always be achieved. Therefore, our algorithm can find a solution for any CSS code case, which is an improvement from previous research. For the problem of fault tolerance, flag [39–42] qubits can in principle be used to identify dangerous errors and recover the quantum state, which are beyond the scope of the present study.

The initial example pertains to the toric code, an early-developed CSS code. For a toric code characterized by a distance d, the total number of physical qubits is given by $n = 2d^2$, and it consistently contains $k = 2$ logical qubits. Figure 3 shows the structure of the toric code, where the Pauli Z and X of the two logical qubits are marked by colored lines that pass the physical qubits they act on. In addition, qubits are placed on the edges of the squares, whereas the faces or vertices of the squares are associated with weight-4 Z or X stabilizers. In the following section, different logical operations will be discussed for toric code.

Several methods exist for simultaneously realizing logical Hadamard gates on two qubits. [32,43] An intuitive method is to apply Hadamard gates to physical qubits transversally and then use SWAP gates to recover the structure of the stabilizers. Another method uses elaborate local gates to realize logical operations, skipping the permutation step of the stabilizers. Owing to the geometric symmetry of the toric code, the above methods, which are based on geometric characteristics, can swap the logical Z and X of two logical qubits conveniently at the same time, whereas they cannot keep one logical qubit and realize the Hadamard gate on the other logical qubit. Thus, the advantages of our algebraic method have emerged. Therefore, we choose the logical operation $\bar{H}_1\bar{I}_2$ to demonstrate the capabilities of our algorithm.

According to Eq. (4), the transformation matrix U_{HI} for logical operation $\bar{H}_1\bar{I}_2$ satisfies the condition

$$\begin{pmatrix} \bar{Z}_1 & 0 \\ \bar{Z}_2 & 0 \\ 0 & \bar{X}_1 \\ 0 & \bar{X}_2 \end{pmatrix} U_{HI} = \begin{pmatrix} 0 & \bar{X}_1 \\ \bar{Z}_2 & 0 \\ \bar{Z}_1 & 0 \\ 0 & \bar{X}_2 \end{pmatrix}, \tag{7}$$

which can be combined with the conditions in Eqs. (2) and (3) for the solution. Once the indices of the physical qubits are determined in Fig. 3, the explicit forms of the parity-check matrix H_Z, H_X and logical Pauli operators will be revealed. Therefore, our algorithm can be executed for the final sequence of the physical gates. The specific result for the case $d = 3$ is shown in the SM. Furthermore, we can count the total number of physical gates of the solution either before or after simplification and calculate the decrease in proportion to test the effect of the simplification method. To determine how the gate resource increases with code distance d, we use the power function $N_{\text{gate}} = N_0 d^{\alpha}$ to fit the results: As illustrated in Fig. 4(a), the fitting curves, represented by blue or purple dotted lines, precisely intersect the data points, with the power α equating to 2.51 prior to simplification and 2.28 subsequent to simplification. This indicates that the impact of the simplification algorithm is more pronounced for larger values of d, a conclusion further corroborated by the gray curve depicting the decreasing proportion. Moreover, the gate resource is not expensive: The number n of physical qubits, which is slightly larger than that of stabilizers, is proportional to d^2. The QEC process requires stabilizer measurements to extract syndromes, consuming several times more physical gates than stabilizers. Consequently, the gate numbers for the Clifford operation and a single round of stabilizer measurements exhibit a similar order of magnitude, indicating that the resource consumption of the Clifford operation is within acceptable limits.

Although a scheme for fault tolerance was not designed in detail, a circuit simulation is necessary to confirm the validity of our solution. Figure 4(b) shows the logical infidelity of $\bar{H}_1\bar{I}_2$ under a given noise rate $p = 10^{-4}$ using the Pauli depolarizing model for both single-qubit and two-qubit gates. [44,45] When the code distance increases, the increase in the logical error rate is slower than that of the number of physical gates, which is beneficial for the future design of fault tolerance. In addition, the simulation can provide proportional logical infidelity to varied physical noise rates; therefore, we believe that physical gates without noise truly realize the targeted logical operation.

Another fundamental Clifford gate is the phase gate S. For the usual surface code, the S gate can be realized using an ancilla logical qubit, as well as several logical CNOT and Hadamard gates. [8] However, for a toric code with two logical qubits, identifying the logical gate $\bar{S}_1\bar{I}_2$ that only realizes the S operation on a single logical qubit is challenging. This problem can be solved using our algebraic method. Before the algorithm begins, we must provide

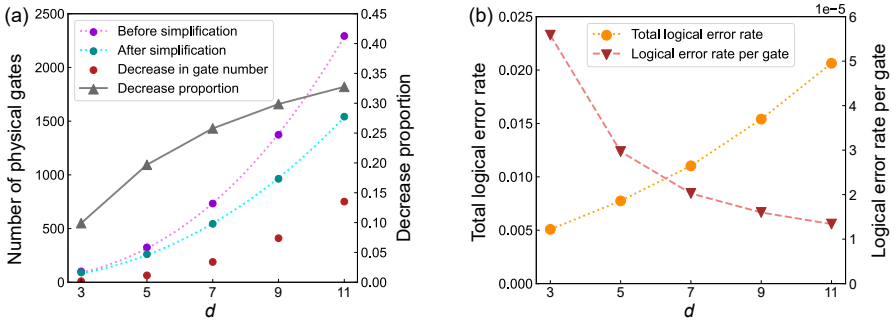

Fig. 4. Resource and performance of the logical operation $\bar{H}_1\bar{I}_2$ for the toric code. (a) Number of physical gates growing with code distance d by the power law $O(d^{2.51})$ before the circuit simplification and $O(d^{2.28})$ after the simplification. (b) Logical error rate of $\bar{H}_1\bar{I}_2$ increasing with d, whereas the averaged error rate per physical gate decreasing with d.

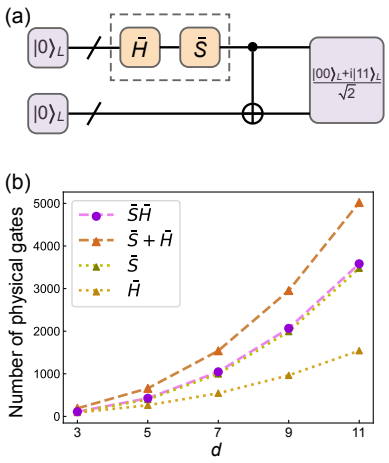

Fig. 5. Example circuit and resources of the logical operation $(\bar{S}_1\bar{H}_1)I_2$ for the toric code. (a) Logical H and S may be encountered in the quantum circuit. (b) Numbers of physical gates for this logical operation are marked by circles and triangles for the one- and two-step schemes, respectively.

the condition for the transformation matrix U_{SI} as

$$\begin{pmatrix} \bar{Z}_1 & 0 \\ \bar{Z}_2 & 0 \\ 0 & \bar{X}_1 \\ 0 & \bar{X}_2 \end{pmatrix} U_{SI} = \begin{pmatrix} \bar{Z}_1 & 0 \\ \bar{Z}_2 & 0 \\ \bar{Z}_1 & \bar{X}_1 \\ 0 & \bar{X}_2 \end{pmatrix}. \tag{8}$$

The results do not yield information of greater significance than that provided by the Hadamard gate. However, the integration of the Hadamard and phase gates may lead to novel insights. Several fundamental Clifford gates may converge within a logical circuit, allowing for the possibility of consolidating some of them into a single operational step. For instance, when logical S and H satisfy the conditions in Eq. (4) written directly for $(\bar{S}_1\bar{H}_1)I_2$, only one sequence of physical qubits can be solved instead of two separate sequences. A simple example is shown in Fig. 5(a). The circuit can be used to produce a phase-shifted Bell state $|\psi\rangle_L = \frac{|00\rangle_L+i|11\rangle_L}{\sqrt{2}}$,[46] which is useful for verifying the fundamental inequalities of quantum entanglement. In Fig. 5(b), the number of physical gates required for $(\bar{S}_1\bar{H}_1)I_2$ is determined following simplification. Thus, the one-step realization can reduce the number of gates by approximately one-third compared to the two-step scheme, with the gate count being only slightly higher than that of the $\bar{S}_1$ step in the two-step scheme. These results demonstrate that our algorithm can efficiently process cases of consecutive logical Clifford operations.

In addition to the toric code that everyone is familiar with, other complicated QEC codes must be analyzed to prove the ability of our algorithm. Here, hyperbolic code is a suitable candidate. Among the commonly used logical Clifford operations, the CNOT gate has received attention in some studies,[25–27,47] in which the Dehn twist is a typical method for determining the sequence of physical gates. However, owing to the complicated symmetry of the hyperbolic code, realizing the Hadamard gate or phase gate of one logical qubit is challenging. For types that are self-dual, some geometrical methods are useful for this target,[48] whereas other cases are troublesome.

Fortunately, the method in the present work does not care if the numbers of Z-type and X-type stabilizers are equal. Therefore, we choose a $\{4,5\}$-tiling hyperbolic code that is not self-dual to demonstrate the calculation of $\bar{H}$ and $\bar{S}$.

Here we briefly introduce the hyperbolic surface code, while its mathematical structure is discussed in SM. Similar to the toric code, the $\{r,s\}$-hyperbolic surface code also places physical qubits on the edges, whereas s neighboring r-gons for Z stabilizers meet at one vertex for an X stabilizer. This connection ensures that each Z stabilizer is r-weighted, and each X stabilizer is s-weighted. The qubits, Z stabilizers, and X stabilizers are all equivalent, which is an important reason for good code performance.

The Poincaré disk, which is a unit disk equipped with a hyperbolic metric,[49,50] is a powerful tool for representing code. On the Poincaré disk, the faces of the $\{r,s\}$-hyperbolic surface code can be selected as the same regular polygons as long as $\frac{1}{r}+\frac{1}{s} < 1/2$ with infinite choices of $\{r,s\}$. However, the Euclidean plane supports only the three tiling methods that satisfy $\frac{1}{r}+\frac{1}{s} = 1/2$. More importantly, although $\{r,s\}$-tiling can be infinitely extended to the boundary of the Poincaré disk, several pairs of edges can be glued to form a part with finite r-gons and finite physical qubits, which can become realizations of the hyperbolic code. When considering the glued component within a three-dimensional space, it forms a closed surface characterized by multiple handles, as opposed to the singular handle associated with the toric code.

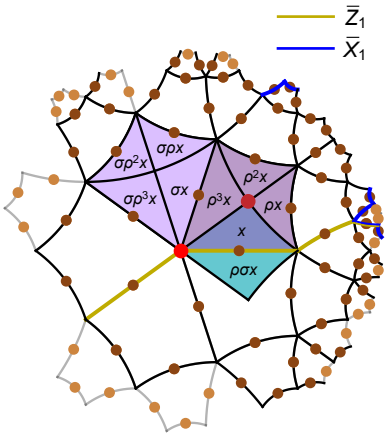

Fig. 6. Schematic for the smallest realization of $\{4,5\}$-hyperbolic surface code. Each physical qubit denoted by a circle is connected to one edge, and one polygon or one vertex represents a Z stabilizer or an X stabilizer, respectively. Owing to the finite size of the lattice for this 60-qubit realization, the boundary of the region needs to be glued to ensure the uniform structure of stabilizers. Therefore, each edge on the boundary is paired with another (light color) for one physical qubit. The Pauli Z and X for one logical qubit are also shown here. Moreover, two red squares and some colored triangles with curved lines are related to the symmetry group of the hyperbolic lattice, which is discussed in SM.

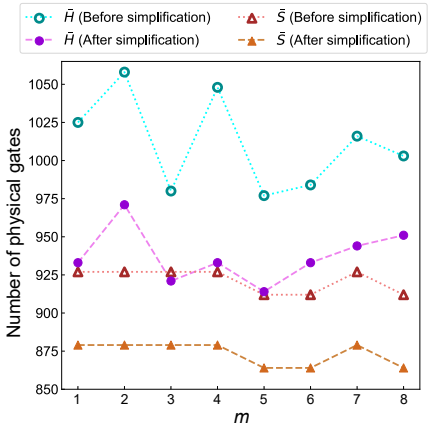

Fig. 7. Resources of the logical Hadamard and phase gates for the hyperbolic surface code. The physical gate numbers of $\bar{H}_m$ and $\bar{S}_m$ are counted for cases of every logical qubit m. The results for different m are calculated separately.

Knowledge of the topology indicates that each handle carries two logical qubits; therefore, the hyperbolic code can encode more logical qubits than the usual toric code. With an increase in the number of physical qubits, the number of logical qubits increases almost proportionally and the code distance increases as well, which shows a high encoding rate.

To demonstrate the capability of our algorithm, we adopt the smallest realization of the $\{4,5\}$-tiling. Using an algebra software named GAP,[51] we find the configuration of 60 physical qubits on the Poincaré disk in Fig. 6. The figure shows 30 Z stabilizers and 24 X stabilizers (both have one redundant stabilizer). The number of logical qubits is 8, calculated as physical qubits minus independent stabilizers. The Pauli Z and X of one logical qubit are marked, sharing one common physical qubit for anti-commutation. Other logical qubits have similar Pauli operators.

As with the toric code, logical Hadamard gates and logical phase gates are calculated for the hyperbolic code, with results shown in Fig. 7. Gate resources remain similar regardless of the operating logical qubit, with a notable reduction after simplification. A comparison with one round of stabilizer measurement using the common circuit,[8] which costs 240 CNOT gates, provides evidence for the effectiveness and generality of the proposed algorithm, whereas QEC requires multiple measurement rounds.

Discussion. In this study, we propose a new method to systematically determine how to realize any targeted Clifford logical operation for any CSS code. The algorithm facilitates the computation of the precise sequence of physical gates, which can be directly implemented by quantum hardware, provided that the QEC module is operational. Our results demonstrate a single logical Hadamard gate and phase gate for the toric and hyperbolic codes, filling the gaps in these logical operations in previous investigations. Our proposed systematic algorithm is expected to be utilized as a powerful tool in the field of LDPC codes.

For practical considerations, circuits for logical operations must be elaborately designed to ensure fault tolerance. Some flag qubit schemes[41,42] can be applied to circuits to recognize correlated errors, and extra physical gates can be added to correct the errors. Several pairs of flag qubits can be inserted into the calculated circuits to identify the dangerous errors. A more careful design can maintain the code distance to achieve a better fault tolerance.

Future simulations of fault-tolerant logical operations across various codes will yield comprehensive insights into circuit performance. Should break-even points with moderate values for fault-tolerant logical operations consistently be identified, the experimentation with LDPC codes will experience significant advancement.

Acknowledgments This study was supported by the National Natural Science Foundation of China (Grant Nos. 12474486, 12234002, and 92250303) and the National Key Research and Development Program of China (Grant No. 2024YFA1612101).

References

[1] Shor P W 1997 *SIAM J. Sci. Comput.* **26** 1484
[2] Zhou L, Wang S T, Choi S, Pichler H, and Lukin M D 2020 *Phys. Rev. X* **10** 021067
[3] Steane A M 1996 *Phys. Rev. Lett.* **77** 793
[4] Bombin H and Martin-Delgado M A 2006 *Phys. Rev. Lett.* **97** 180501
[5] Ryan-Anderson C, Bohnet J G, Lee K *et al.* 2021 *Phys. Rev. X* **11** 041058
[6] Postler L, Heußen S, Pogorelov I *et al.* 2022 *Nature* **605** 675
[7] Ryan-Anderson C, Brown N C, Baldwin C H *et al.* 2024 *Science* **385** 1327
[8] Fowler A G, Mariantoni M, Martinis J M, and Cleland A N 2012 *Phys. Rev. A* **86** 032324
[9] Krinner S, Lacroix N, Remm A *et al.* 2022 *Nature* **605** 669
[10] Acharya R, Aleiner I, Allen R *et al.* 2023 *Nature* **614** 676

[11] Litinski D 2019 *Quantum* **3** 128
[12] Bluvstein D, Evered S J, Geim A A *et al.* 2024 *Nature* **626** 58
[13] Heußen S, Locher D F, and Müller M 2024 *PRX Quantum* **5** 010333
[14] Breuckmann N P and Eberhardt J N 2021 *PRX Quantum* **2** 040101
[15] Strikis A and Berent L 2023 *PRX Quantum* **4** 020321
[16] Bravyi S, Cross A W, Gambetta J M *et al.* 2024 *Nature* **627** 778
[17] Sahay K and Brown B J 2022 *PRX Quantum* **3** 010310
[18] Zhu G, Jochym-O'Connor T, and Dua A 2022 *PRX Quantum* **3** 030338
[19] Schotte A, Zhu G, Burgelman L, and Verstraete F 2022 *Phys. Rev. X* **12** 021012
[20] Vasmer M and Kubica A 2022 *PRX Quantum* **3** 030319
[21] Bruzewicz C D, Chiaverini J, McConnell R, and Sage J M 2019 *Appl. Phys. Rev.* **6** 021314
[22] Bravyi S, Poulin D, and Terhal B 2010 *Phys. Rev. Lett.* **104** 050503
[23] Baspin N and Krishna A 2022 *Phys. Rev. Lett.* **129** 050505
[24] Breuckmann N P and Terhal B M 2016 *IEEE Trans. Inf. Theory* **62** 3731
[25] Breuckmann N P, Vuillot C, Campbell E *et al.* 2017 *Quantum Sci. Technol.* **2** 035007
[26] Lavasani A, Zhu G, and Barkeshli M 2019 *Quantum* **3** 180
[27] Zhu G, Lavasani A, and Barkeshli M 2020 *Phys. Rev. B* **102** 075105
[28] Calderbank A R, Rains E M, Shor P M, and Sloane N J A 1998 *IEEE Trans. Inf. Theory* **44** 1369
[29] Bravyi S, Terhal B M, and Leemhuis B 2010 *New J. Phys.* **12** 083039
[30] Aaronson S and Gottesman D 2004 *Phys. Rev. A* **70** 052328
[31] Sayginel H, Koutsioumpas S, Webster M *et al.* 2025 *PRX Quantum* **6** 030343
[32] Kobayashi R and Zhu G 2024 *PRX Quantum* **5** 020360
[33] Patel K N, Markov I L, and Hayes J P 2008 *Quantum Inf. Comput.* **8** 282
[34] Bataille M 2022 *Quantum Inf. Process.* **21** 269
[35] Maslov D, Dueck G W, Miller D M, and Negrevergne C 2008 *IEEE Trans. Comput. Aided Des. Integr. Circ. Syst.* **27** 436
[36] Dennis E, Kitaev A, Landahl A, and Preskill J 2002 *J. Math. Phys.* **43** 4452
[37] Kitaev A Y 2003 *Ann. Phys.* **303** 2
[38] Gidney C 2021 *Quantum* **5** 497
[39] Chao R and Reichardt B W 2018 *Phys. Rev. Lett.* **121** 050502
[40] Chao R and Reichardt B W 2018 *npj Quantum Inf.* **4** 42
[41] Chao R and Reichardt B W 2020 *PRX Quantum* **1** 010302
[42] Chamberland C and Beverland M E 2018 *Quantum* **2** 53
[43] Barkeshli M, Chen Y A, Huang S J *et al.* 2023 *SciPost Phys.* **14** 065
[44] Trout C J, Li M, Gutiérrez M *et al.* 2018 *New J. Phys.* **20** 043038
[45] Raussendorf R and Harrington J 2007 *Phys. Rev. Lett.* **98** 190504
[46] Davis J J J, Jackman C L, Leonhardt R *et al.* 2025 *J. Opt. Soc. Am. B* **42** 1227
[47] Lavasani A and Barkeshli M 2018 *Phys. Rev. A* **98** 052319
[48] Breuckmann N P and Burton S 2024 *Quantum* **8** 1372
[49] Levy S 1997 *Flavors of Geometry* (Cambridge University Press)
[50] Boettcher I, Gorshkov A V, Kollár A J *et al.* 2022 *Phys. Rev. B* **105** 125118
[51] GAP – Groups, Algorithms, and Programming, Version 4.13.1 2024 https://www.gap-system.org