Construction of color codes from polygons

To cite this article: Eduardo Brandani da Silva and Waldir S Soares Junior 2018 J. Phys. Commun. 2 095011

View the article online for updates and enhancements.
Construction of color codes from polygons

Eduardo Brandani da Silva$^{1,3}$ and Waldir S Soares Junior$^2$

$^1$ Department of Mathematics, Maringa State University Av. Colombo, 5790 Maringa, PR Brazil 87020-900
$^2$ UTFPR—Pato Branco Via do Conhecimento, km 1 Pato Branco—PR, Brazil 85503-390
$^3$ Author to whom any correspondence should be addressed.
E-mail: ebsilva@uem.br and waldirjunior@utfpr.edu.br

Keywords: quantum color codes, triangular color codes, quantum error correction, surface codes

Abstract
We propose a procedure for constructing new classes of codes, called polygonal color codes. This technique expands the triangular color codes introduced by Bombin and Martin-Delgado in 2007. Triangular color codes, built on a two-dimensional Euclidean surface with three edges, implement the entire Clifford group, but they only encode one qubit. In our case, polygonal color codes are constructed on two-dimensional surfaces with $n$ edges, with $n \geq 3$, increasing the number of encoded qubits. Considering a surface with boundary and evaluating the generators of the first homology group of this surface, we may increase the dimension of the code without losing the transversal implementation of the Clifford group. In addition, we construct two codes, one with 2 and one with 3 encoded qubits, which exemplify the method and show that we can generate codes with better parameters than the known triangular codes. For one of these constructions, we also give general parameters.

1. Introduction

In 1982, Richard Feynman theoretically proposed a computer based on the principles of quantum mechanics [1]. The concept was modified over the years and, after the description of a quantum computer given by Deutsch [2], there was a great advance in the research of this area.

One of the most famous results was presented by Peter Shor in 1994. The Shor algorithm showed that, using quantum properties, a quantum computer could factor a number with $n$ digits with an amount of steps growing geometrically. In a conventional computer this amount grows exponentially [3].

One of the great difficulties of performing quantum computing is decoherence, as Unruh warned in 1995 [4]. Decoherence is the decay phenomenon of superposition of states, due to the interaction between the system and the surrounding environment. In theory, the problem may be solved using quantum error-correcting codes.

Classic error-correcting codes were introduced by Claude Shannon in 1948 [5] and inspired the creation of quantum error-correcting codes. The first code, introduced by Shor in 1995 [6], was actually a concatenation of two 3-qubit codes, one that protected against bit-flip errors and one that protected against phase-flip errors, the latter without any classic analogue.

In 1996 there was a breakthrough in quantum coding due to the discovery of a class of codes, now known as CSS codes, by Robert Calderbank, Peter Shor and Andrew Steane [7, 8]. These codes were generalized in a rich code structure, which are the stabilizer quantum codes [9].

A powerful technique for quantum correction of errors is given by the toric codes, proposed by Kitaev [10], giving origin to the topological quantum error-correcting codes. One of the great advantages of these codes is that their parity check operators are geometrically local, that is, they always act in a small amount of qubits in their neighborhood. Quantum words are therefore intrinsically resilient to local noises, which do not affect the topological properties of the system [11].

Toric codes has been generalized and somewhat improved, as may be seen in the researches by Bombin and Martin-Delgado, [12], by Albuquerque, Pallazo and Silva [13–15], by Delfosse, [16], Terhal and Breuckmann, [17], Silva and Soares [18], among many others.
All these codes transversely implement the quantum gates CNOT, $X$ and $Z$. In order to increase the implementable quantum gates and thus increase the number of tasks performed, a different class of codes has been proposed in [19]: the Color Codes. Under appropriate circumstances, color codes can implement transversely the whole Clifford group.

The Clifford group is the second level of the so-called Clifford hierarchy (CH), introduced by Gottesman and Chuang [20] for quantum operations, and defined by

$$\mathbb{P}_t = \{ U | UPU^\dagger P^\dagger \in \mathbb{P}_t, \forall P \in \mathbb{P}_t \},$$

where $U$ is a unitary operator acting on $n$ qubits. In this definition, the first level of the hierarchy is $\mathbb{P}_2 = \mathbb{P}_n$, which is the Pauli group defined in the section 3. The second level is the Clifford group, which is the normalizer of the Pauli group. To generate the entire Clifford group we need only three operators, which are $H$ (Hadamard gate), $K$ (phase shift gate, or $\pi/4$ gate) and $\Lambda(X)$ (controlled-not gate or CNOT gate):

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad K = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad \Lambda(X) = \begin{pmatrix} 1 & 0 \\ 0 & X \end{pmatrix}.$$  

Although the implementation of the Clifford group does not imply the universality of quantum computation (which we would have at the third level of CH), there are important tasks performed by the operations of this group, such as quantum distillation, quantum teleportation and dense coding, for instance. There are other works that propose different techniques for topological coding that also allow the implementation of the Clifford group, as we may see in [21] and [22].

The strength of the color codes was demonstrated when an example of this code was implemented in 2014 [19]. The authors describe the implementation of a quantum error-correcting code encoding one qubit in entangled states distributed over 7 trapped-ion qubits. They demonstrate the capability of the code to detect one bit flip, phase flip or a combined error of both, regardless on which qubit they occur.

Triangular color codes were introduced in 2007. Kubica, Yoshida and Pastawski [24] proved that color codes on a $d$-dimensional closed manifold is equivalent to multiple decoupled copies of $d$-dimensional toric codes, up to local unitary transformations, adding or removing ancilla qubits. Moreover, for a $d$-dimensional color code with $d + 1$ boundaries of $d + 1$ distinct colors, they obtained that the code is equivalent to multiple copies of $d$-dimensional toric codes which are attached along a $(d - 1)$-dimensional boundary.

Anderson [25] defined the notion of label set equivalencies and showed that under a small set of constraints, the only homological stabilizer codes without local logical operators are equivalent either to Kitaev’s toric code or to the topological color codes. He also discuss how many logical operators exist in a code with $n$ boundaries, with or without holes.

In [26], Bravyi and Kitaev discuss the technique of increasing the number of edges for surface codes. Cesare [27] evaluated the increase in the number of qubits encoded in color codes using the introduction of defects on a two-dimensional surface. More recently, several works were generalized by Delfosse, Iyer and Poulin [28], where they consider a notion of relative homology (and cohomology) for surfaces with mixed boundaries, considering a construction which includes both Bravyi and Kitaev’s and also Freedman and Meyer’s [29] extension of Kitaev’s toric code.

In current paper, we propose a construction of color codes on a bidimensional surface with multiple borders, thus obtaining the polygonal color codes $\mathcal{P}(n, m)$, which are a generalization of the triangular color codes of Bombin and Martin-Delgado. These codes preserve the implementation of the whole Clifford group.

To obtain these codes we increase the number of edges in the boundary of the surface and evaluate the behavior of the strings. We consider both direct strings with a single color, or $t$-strings, which are strings that are equivalent deformed by combining two colors to obtain a third one.

The main result is that we increase the number of coded qubits, but the elements that generate the homology group are the $t$-strings, which makes possible to implement the Clifford group as in the triangular color code, and we can still generate codes with better parameters than the triangular ones. We also construct two examples of codes, one with 2 and one with 3 encoded qubits, which show the method and that we can generate codes with better parameters than the known triangular codes. For one of these constructions, we also give general parameters.

2. Homology

The topological quantum codes that we are going to work with depend on the homology of the surface. Let us go through the introductory concepts of homology and the first group of homology of a space or a surface. For more details see [30–32].

Since we are going to work on a very specific case, we will not deal with definitions of homology on any topological space, but on two-dimensional surfaces.
By embedding a graph $G$ on surface $S$ we say that we are doing a cellulation of the surface. The vertices of $G$ are called 0-cells, the edges are 1-cells and the faces 2-cells.

**Definition 1.** Given a graph $G$ embedded on a surface $S$, let $P$ be the set of all $p$-cells. For each $P' \subset P$, we define the $p$-chain in $S$ be the finite sum

$$c = \sum_i c_i p_i, \text{ such that } c_i = \begin{cases} 0; & p_i \notin P' \\ 1; & p_i \in P' \end{cases}.$$  

In particular, if we label the edges of a cellulation of the surface by $\{e_i\}_{E}$, we may represent the 1-chains defined above by the following formal sum:

$$c = \sum_i c_i e_i, \text{ such that } c_i = \begin{cases} 0; & e_i \notin E' \\ 1; & e_i \in E' \end{cases}$$

where $E' \subset P$ is a set of edges of $G$.

Further, 1-chains may be summed module 2, producing an Abelian additive group $C_1$. Similarly, 0-chains and 2-chains of $G$ will determine the additive Abelian groups $C_0$ and $C_2$, respectively.

We relate groups of $p$-chains using edge operators, which we denote by $\partial_i : C_i \rightarrow C_{i-1}$, $i = 0, 1, 2$. These operators send an element on their border. For example, if $f$ is a face of $G$ whose boundary is given by edges $\{e_1, \ldots, e_k\}$, then $\partial_2(f) = e_1 + \cdots + e_k$. A $p$-chain $c$ is called a cycle if $\partial_p c = 0$. In figure 1 we have instances of these actions.

According to sequence $C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \rightarrow \{0\},$

we note that $\text{Im}(\partial_{i+1}) \subset \ker(\partial_i)$.

Let us define two subgroups of $C_1$, which we denote by $Z_1$ and $B_1$. The subgroup $Z_1$ is the kernel of $\partial_1$, that is, the sets of edges (1-chains) that form a cycle, a closed curve. The subgroup $B_1$ is the image of $\partial_2$, that is, the set of edges that form the boundary of a 2-chain.

Now we may define:

**Definition 2.** The first homology group of a surface $S$ is the quotient

$$H_1 = \frac{Z_1}{B_1}$$

In general, the first homology group is a finitely generated Abelian group and, regardless of the graph used in the embedding, the number of generators of the first homology group has much to say about the topological nature of the surface. The number of generators of the first homology group is twice the genus of the surface considered, as may be seen in [31] and [32].
For instance, in a torus let $a$ and $b$ be the generators of the first homology group. We remark that in the torus there are infinitely different closed curves, but they may be deformed continuously for curve $a$ or for curve $b$, or for a concatenation of $a$ with $b$, or even for a single point (the neutral element 0 of the group), which we call a homologically trivial curve. In figure 2 is shown these curves.

In this work we need not only the concept of homology of a surface, but also the concept of relative homology. As already mentioned, these concepts have already been introduced and used in several previous works on similar contexts, such as in [19, 26–28].

3. Color codes

A quantum error-correcting code (QECC) is an application of a Hilbert space $H^k$, of dimension $2^k$, to a Hilbert space of dimension $2^n$ with $k < n$. A QECC $C$ with code word length $n$, dimension $k$, and minimum distance $d$ is denoted by $[[n, k, d]]$.

The first classes of quantum error-correcting codes were those that dealt with bit-flip and phase-flip errors or a combination of them, such as the Shor code [6], proposed in 1995.

On the other hand, the color codes introduced by Bombin and Martin Delgado in [19] belong to a class of codes called stabilizer quantum codes. Stabilizer codes are the quantum analogue for the classic linear codes [33]. They are defined by the stabilizer group.

**Definition 3.** Let $P = \{I, X, Y, Z\}$ be the set of the Pauli matrices on 1 qubit. The Pauli group on $n$ qubits is the set $P^n$ of elements in the form $i^jP_1 \otimes P_2 \otimes \ldots \otimes P_n$, where for every $j = 1, \ldots, n$ we have $P_j \in P$ and $k \in \{0, 1, 2, 3\}$. If we have an Abelian subgroup $S \subset P^n$ such that $-1 \not\in S$, we call $S$ the stabilizer group.

Thus, the stabilizer code $C$ is defined as the eigenspace associated with the eigenvalue 1 of the operators in $S$, that is: [9]

$$C = \{|\psi\rangle \in H^n; M|\psi\rangle = |\psi\rangle, \forall M \in S\}.$$

If a stabilizer code has $r$ generators, then the dimension of the stabilizer subspace is $2^{n-r}$ [33], implying that the number of coded qubits is:

$$k = n - r \tag{1}$$

Kitaev [10] proposed a particular case of stabilizer codes, the toric codes. To define them, we consider a tessellation by squares in the flat torus $l \times l$. The edges of this tessellation were placed in a one-to-one correspondence with the qubits. Since each edge belongs to two faces of the tessellation, we have $|E| = 2l^2$, that is, the length of the code is $n = 2l^2$.

The stabilizer operators are associated with the vertices and faces of the tessellation. Operator $X_v$, associated to each vertex $v$ of the tessellation, is given by the tensor product of Pauli matrices $X$ and identity matrices, where we have a Pauli matrix $X$ at each edge adjacent to the vertex, and the identity at all other edges. Each face $f$ of the tessellation is associated with operator $Z_f$, given by the tensor product of Pauli matrices $Z$ and identity matrices, where we have a Pauli matrix $Z$ at each edge on the border of the face, and identity matrices at all other edges of the faces. In short:

$$X_v = \bigotimes_{j \in E} X^{x_{j}(j \in E_v)}, \quad Z_f = \bigotimes_{j \in E} Z^{z_{j}(j \in E_f)}$$
Two examples of operators is given in figure 3. Kitaev toric code is defined as:

\[ C = \{ |\psi\rangle; X_v |\psi\rangle = |\psi\rangle; Z_f |\psi\rangle = |\psi\rangle; \forall v, f \}, \]

Kitaev code has parameters \([2^k, 2, l]\), where \(k = 2\) is the number of homologically non-trivial cycles and \(d = l\) is the number of edges contained in the shortest homologically non-trivial cycle.

This code is one of the most important classes of quantum error-correcting codes. It was generalized to the surface codes and homological codes. In addition, studies of these codes were also done on surfaces with borders, with mixed edges or with holes, in order to modify the group of homology of the considered surface, and thus to generate new families of codes, as can be seen in the works [24, 26, 28].

As mentioned above, the color codes, introduced by Bombim and Martin-Delgado in [19], are a particular case of stabilizer quantum codes and, similar to the Kitaev code, is a topological code.

To generate a color code on a two-dimensional surface, a trivalent, 3-colorable tessellation is required. Trivalent tessellation means that, at each vertex of the tessellation, exactly 3 edges meet. A tessellation is 3-colorable if it is possible to color all the faces of the tessellation using only 3 colors (we fix red, green and blue), so that two faces sharing an edge do not have the same color. Considering a coloration of faces, we also induce a coloration of the edges, so that an edge of a certain color does not belong to the border of a face with the same color.

In color codes, the qubits are associated one-to-one with the vertices of the tessellation. The generators of the stabilizers are the face operators, with both \(X\) and \(Z\) operators on each face, acting on the vertices of the face in question. For each face \(p\) of the tessellation, we will denote such operators by \(B_p^\sigma\) with \(\sigma = X, Z\).

Separating the faces according to their color, in the sets \(R\) (red), \(G\) (green) and \(B\) (blue), we have:

\[ \prod_{p \in R} B_p^X = \prod_{p \in G} B_p^Z = \prod_{p \in B} B_p^Z \]  \(\text{(2)}\)

with \(\sigma = X, Z\).

Equation (2) implies that four of these generators are not independent and, as we have seen in (1), we may calculate the number of encoded qubits. Since the number of generators is \(r = 2 (V + F - 2)\) and the number of physical qubits is double the number of edges \((n = 2E)\), one has [11]

\[ k = n - r = 4 - 2X \]  \(\text{(3)}\)

where \(X\) is the Euler characteristic of the surface. Note that the number of encoded qubits \(k\) depends only on the surface considered, but not on the tessellation used.

It is appropriate to introduce the concept of shrunk lattices, one for each color, which are auxiliary lattices. For instance in the red shrunk lattice we place a vertex on each red face. The edges of the reduced net correspond to two vertices of the original whilst the green and blue faces of the colored tessellation are the faces of that auxiliary lattice.

To be clear on how color codes work, we need to understand how string operators work. As in the toric codes, the homology of these strings is defined on \(\mathbb{Z}_2\), since we are dealing with two-level quantum systems.

These strings may be green, blue or red, depending on which net we are considering. Regardless of the color, they can be of type \(X\) or \(Z\). Let us denote these string operators by \(S_{\mu}^{C,\sigma}\) where \(C\) is a color, \(\sigma \in \{Z, X\}\) and \(\mu\) is a label of the class of homology.
In general, one has [19]:

\[ S^B \cdot S^G \cdot S^R \sim 1 \] (4)

Equation (4) shows that there are only two independent colors.

A fundamental property of color codes is that, in addition to a string operator of one color being able to be deformed to another homologous string operator, it is also possible to combine two strings of different colors to produce a third equivalent operator of different color from the two original ones. Strings and t-strings are showed in figure 4.

There are good color codes generated on compact surfaces, as we may see in [17] or [18], for example. But on surfaces with bondaries, things take a different direction.

According to the construction in [19], a color code was created on a planar region bounded by a triangle in which each edge of the triangle is one border, each of one color. The code encodes only one qubit, and its non-trivial homology string is a t-string, that is, a string that departs from one edge of a certain color, branches into two other colors, and each part of its branching ends in the border which has its color, see figure 5.

The t-strings in the triangular codes are the key to implement the Clifford group. The difference in relation to the surface codes is that, geometrically, the t-strings X and the t-strings Z are exactly the same. Thus, if we denote two such t-strings by \( T^X \) and \( T^Z \), we have \( (T^X, T^Z) = 0 \).

Thus, given the encoded operators \( \bar{X} = X^{\otimes n} \), \( \bar{Z} = Z^{\otimes n} \), \( \bar{K} = K^{\otimes n} \) and \( \bar{H} = H^{\otimes n} \), where \( n \) is the number of qubits in the code, we have

\[ \bar{H}.X_f.\bar{H} = Z_f, \quad \bar{H}.Z_f.\bar{H} = X_f, \quad \bar{H}.X.\bar{H} = Z \quad \text{e} \quad \bar{H}.Z.\bar{H} = \bar{X}. \]

Further,

\[ \bar{K}.Z_f.\bar{K}^+ = Z_f \quad \text{and} \quad \bar{K}.X_f.\bar{K}^+ = (-1)^{f/2}X_f.Z_f, \] (5)

where \( f \) is the number of vertices of the face. Thus, using a tessellation where all faces have a number of vertices multiple of 4, such as \{4, 8, 8\} (the only one possible in the Euclidean plane), all generators of the Clifford group
may be implemented transversally, since the CNOT gate is always implemented transversally in CSS codes [11, 19, 33].

4. Multiple borders

Color codes are error-correcting codes with good parameters when applied to compact surfaces. But when applied to the surfaces with boundary, as the triangular color codes, we have a fundamental property, which is the possibility of implementing the entire Clifford group, allowing to perform several important tasks in the Quantum computation.

The main drawback is that the triangular codes encode only one qubit. Thus, if we increase the number of encoded qubits, we will improve the parameters of the codes, and save resources for their implementation.

Our main objective is to construct color codes with the good properties of the triangular case, but with the implementation of more than one qubit. To attain this goal, we consider polygons with a number of sides greater than three. We consider a proper way to color the edges and we will verify which are the strings of non-trivial homology that generate the others, in each analyzed case.

We fix a color for only one edge and alternately increasing the amount of edges for each one of the other two colors considered. Without loss of generality, we always fix the blue color in only one edge and increase the amount of green and red edges, one by one, and in this order.

The first case is the triangular codes of Bombim and Martin-Delgado, denoted by $P(1, 1, 1)$ (1 blue—1 green—1 red).

The second case is a quadrilateral, where we only have one blue border (always), two green and one red, which we denote by $P(1, 2, 1)$, or Polygonal Code 1–2–1.

Following the rules to strings of a surface with labeled edges, in our case by colors, a cycle can be homologically non-trivial if it initiates and ends in distinct edges of the same color; or, in the case of a $t$-string it initiates on one edge of a certain color, passes through a bifurcation point and, from that point, each of the two generated strings ends at an edge of its corresponding color. Thus, in the quadrilateral described previously, we may have, except for continuous deformations, three types of cycles with non-trivial homology: a cycle that initiates from one green edge and ends at another green edge, and two $t$-strings initiating at one of the green edges, branches and ends at the corresponding edges, as in figure 6.

However, we may observe that one of these strings may be generated by the product (concatenation) of the other two, namely, the green string may be generated by the two $t$-strings, as shown in figure 7. This means that the two $t$-strings in question generate any homologically non-trivial cycle of this surface. Thus, the code $P(1, 2, 1)$ encodes two qubits maintaining an important property: since both generator cycles are $t$-strings, the code generated on the surface will still implement the entire Clifford group transversely, since it has the same essential properties of the triangular codes described in section 3.

Following the established technique for growth of the number of sides of the polygons, for the third case consider a polygon with 5 sides of the form $P(1, 2, 2)$. We enumerate the edges in order to facilitate the notation, taking the blue border as base and going through the polygon clockwise, we have the green edges $G_1$ and $G_2$ and the red edges $R_1$ and $R_2$. Similar to the quadrilateral, there are six cycles of non-trivial homology: a green cycle ($g_{12}$) which starts at $G_1$ and ends at $G_2$, a red cycle ($r_{12}$) which starts at $R_1$ and ends at $R_2$, and the four $t$-strings
(\(T_{11}, T_{12}, T_{21}, T_{22}\)), where we denote by \(T_{ij}\) the string starting at the blue edge, branches and ends on the sides \(G_i\) and \(R_j\), as shown in figure 8.

Again, although we have six possibilities for non-trivial homology strings, some of these strings can be generated by the others. Observe the following systems:

\[
\begin{align*}
T_{11} + T_{21} &= \eta_2 \\
T_{12} + T_{22} &= \eta_2 \\
T_{21} + T_{22} &= \eta_2
\end{align*}
\]

We can observe that, as in the previous case, direct strings of a single color can be generated by \(t\)-strings. In addition, we can also generate a \(t\)-string in function of others. For instance, from the first system we have that \(T_{11} + T_{21} = T_{12} + T_{22}\), which implies that \(T_{11} + T_{21} + T_{12} = T_{22}\). The same conclusion can be obtained through the second system.

Therefore, from the six non-trivial classes of homology, we have that only three of them generate all the others, in this case \(T_{11}, T_{12}, T_{21}\).

We could have chosen another combination of generators, including some combination that included one or more direct strings, but a choice with only \(t\)-strings is the best for our purposes. Generalizing this idea, we have the following result.

**Theorem 4.** Let \(P\) be a polygon with \(n + m + 1\) sides, where \(m = n\) or \(m = n + 1\). Then, we can generate the \(P(1, m, n)\) code that encodes \((m + n - 1)\) qubits so that the encoded qubits are always represented by \(t\)-strings.

The hypothesis that \(m = n\) or \(m = n + 1\) guarantees that it is possible to construct the polygon without adjacent sides having the same color.

In the case of \(m = n\), we have a polygon with an odd number of sides, so we fix the blue side (the only one). A sequence of \(n\) edges, whose colors alternate between green and red, starting with red, leaves from one of the
vertices of this edge. Another sequence of edges, whose colors alternate between red and green, starting with green, starts from the other vertex of the blue side. Since both sequences have the same number of edges and begin with different colors, they end in different colors. The last two sides of each sequence join together by closing the polygon.

On the other hand, if \( m = n + 1 \), we have a polygon with an even number of sides. Again we fix the blue side. From each one of its vertices we have a sequence of \( n \) edges, where both sequences initiate with the green side. If \( n \) is even, there is the same number of red and green edges in each sequence, and both end in red. The last green side is used to connect the two red edges to close the polygon. If \( n \) is odd, there is one more green edge than the amount of red edge in each sequence of edges, and both sequences end in green. The last red edge connects the last two green edges to close the polygon.

In relation to the number of encoded qubits, we again use the enumeration of the edges as already done previously. So we have \( m \) green edges \((G_1, ..., G_m)\) and \( n \) red edges \((R_1, ..., R_n)\).

For each direct green string, we can generate a system of equations that represent strings equivalences, then we have, starting from the edge \( G_1 \):

\[
\begin{align*}
T_{i1} + T_{i2} &= g_{i2} \\
T_{i2} + T_{i3} &= g_{i3} \\
&\vdots \\
T_{in} + T_{2n} &= g_{i2} \\
T_{11} + T_{31} &= g_{13} \\
T_{12} + T_{32} &= g_{13} \\
&\vdots \\
T_{1n} + T_{3n} &= g_{13} \\
T_{11} + T_{m1} &= \tau_{m1} \\
T_{12} + T_{m2} &= \tau_{m2} \\
&\vdots \\
T_{1n} + T_{mn} &= \tau_{mn}
\end{align*}
\]

Starting from the edge \( G_2 \):

\[
\begin{align*}
T_{i1} + T_{i2} &= g_{i2} \\
T_{i2} + T_{i3} &= g_{i3} \\
&\vdots \\
T_{in} + T_{2n} &= g_{i2} \\
T_{21} + T_{41} &= g_{24} \\
T_{22} + T_{42} &= g_{24} \\
&\vdots \\
T_{2n} + T_{4n} &= g_{24} \\
T_{21} + T_{m1} &= \tau_{m1} \\
T_{22} + T_{m2} &= \tau_{m2} \\
&\vdots \\
T_{2n} + T_{mn} &= \tau_{mn}
\end{align*}
\]

And so on until the side \( G_{m-1} \):

\[
\begin{align*}
T_{(m-1)i1} + T_{m1} &= g_{(m-1)m} \\
T_{(m-1)i2} + T_{m2} &= g_{(m-1)m} \\
&\vdots \\
T_{(m-1)in} + T_{mn} &= g_{(m-1)m}
\end{align*}
\]

Similarly, for each direct red string, we can also generate a system of equations representing equivalent strings, starting from the edge \( R_1 \):

\[
\begin{align*}
T_{i1} + T_{i2} &= \tau_{i2} \\
T_{i2} + T_{i3} &= \tau_{i3} \\
&\vdots \\
T_{in} + T_{2n} &= \tau_{i2} \\
T_{1i} + T_{3i} &= \tau_{13} \\
T_{2i} + T_{3i} &= \tau_{23} \\
&\vdots \\
T_{ni} + T_{3n} &= \tau_{13} \\
T_{11} + T_{ni} &= \lambda_{ni} \\
T_{12} + T_{ni} &= \lambda_{ni} \\
&\vdots \\
T_{1n} + T_{ni} &= \lambda_{ni}
\end{align*}
\]

Starting from the edge \( R_2 \):

\[
\begin{align*}
T_{i2} + T_{i3} &= r_{i3} \\
T_{i2} + T_{i4} &= r_{i4} \\
&\vdots \\
T_{in} + T_{2n} &= r_{i3} \\
T_{2i} + T_{ni} &= \lambda_{ni} \\
T_{2i} + T_{ni} &= \lambda_{ni} \\
&\vdots \\
T_{2n} + T_{ni} &= \lambda_{ni}
\end{align*}
\]

And so it goes until the strings starting from the edge \( R_{n-1} \):

\[
\begin{align*}
T_{(n-1)i1} + T_{in} &= \lambda_{(n-1)n} \\
T_{(n-1)i2} + T_{2n} &= \lambda_{(n-1)n} \\
&\vdots \\
T_{(n-1)in} + T_{mn} &= \lambda_{(n-1)n}
\end{align*}
\]

Taking the \( t \)-strings in the first row from each system, we obtain \( n \) \( t \)-strings from the systems of the green strings, and \( m \) \( t \)-strings from the systems of red strings, where the \( t \)-string \( T_{11} \) appears in both groups, which results in \( m + n - 1 \) \( t \)-strings \((T_{11}, T_{12}, ..., T_{1m}, T_{21}, T_{22}, ..., T_{2n}, T_{31}, ..., T_{m1})\).

It is easy verify that all strings, both direct or \( t \)-strings, can be generated from these \((m + n - 1)\) \( t \)-strings, taking in account all the equations in the systems presented.

The transversal implementation of the Clifford group by these codes comes from the same facts already reported for the first case, the triangular code. The transverse action of the Hadamard gate \( H \) is equivalent to a
Hadamard gate over the coded qubits, precisely by the nature of the $X$-strings and $Z$-strings. When we apply transversally $H$ in a t-string of type $X$ we have $HTXZ = T$, and the resulting t-string, which is of type $Z$ has the same support as $X$, containing an odd number of qubits, so they anticommute. And for the gate $K$, the transversal implementation requires an extra constraint, as we may see in equation (3), which is solved using a convenient tessellation, in the case the $\{4, 8, 8\}$.

5. New families of color codes

We give examples of codes of type $P(1, m, n)$, initially focusing on $P(1, 2, 1)$, which we give a general construction. These codes have the same properties of $P(1, 1, 1)$ (which is the triangular code of Bombin and Martin-Delgado) but present better parameters.

We begin considering a quadrilateral and we use the semiregular tessellation $\{4, 8, 8\}$, which is the only one possible in the Euclidean plane with all polygons having the number of sides multiple of four. We note that the minimum distance of the code can be made so great as we want.

In the example of figure 9 we use a rectangle of measurements $(2 + \sqrt{2}) \times (1 + \sqrt{2})$ and the polygons of the tessellation with sides of length 1, thus generating a code with parameters $[12, 2, 3]$.

Comparing this code to the simpler triangular code in [19], we can see that both have distance 3, but the coding rate of $P(1, 2, 1)$ is $\frac{1}{6}$ while in the triangular is $\frac{1}{7}$.

By successively increasing both the length and the width of the rectangle by $2 + \sqrt{2}$ the minimum distance of the code increases by 2. Then, it is possible to have control of the growth of the number of qubits contained in the codes. Figure 10 exemplifies two of these steps. Thus, by this procedure we can generate a family of codes of type $P(1, 2, 1)$ with parameters $[8m^2 + 2m + 2, 2m + 1]$, $m \geq 1$.

Comparing with the triangular code $[73, 1, 9]$ given by Bombin and Martin-Delgado in [33], the $P(1, 2, 1)$ code of the family described above with the same minimum distance is the $[138, 2, 9]$, whose coding rate is $\frac{4}{73}$, while that of the triangular case is $\frac{1}{73}$, showing that the use of the polygonal codes can present gains in relation to the triangular ones.

In the example of figure 11 we use a pentagon and generate a code with parameters $[161, 3, 9]$. It is worth noting that this code has distance 9, as well as the triangular code $[73, 1, 9]$ presented by Bombin and Martin-Delgado in [33], but if we observe the coding rate of both, we have that the rate of the $P(1, 2, 2)$ code is $\frac{1}{161}$, while in the triangular case, the rate is $\frac{1}{73}$, which shows a real gain in the use of polygonal code in relation to the triangular code.

We call the attention to the fact that in both cases we do not use regular polygons. As we change the measurements of the external polygon which will be tessellated, we generate new families of codes, with better or worse parameters.
6. Conclusion

In this work, we have proposed a procedure for obtaining new classes of color codes, called polygonal color codes. This technique expands the triangular color codes introduced by Bombin and Martin-Delgado. Polygonal color codes are constructed on two-dimensional surfaces with \( n \) edges, with \( n \geq 3 \), increasing the number of encoded qubits. Considering a surface with boundary and evaluating the generators of the first homology group of this surface, we increase the dimension of the code without losing the transversal implementation of the Clifford group.

Future work shall focus on simulating the performance of these codes with respect to different error models.

ORCID iDs

Eduardo Brandani da Silva  
https://orcid.org/0000-0003-2687-5174

Waldir S Soares Junior  
https://orcid.org/0000-0001-6216-8691

References

[1] Feynman R P 1982 Simulating physics with computers Int. J. Theory Physics 21 467

[2] Deutsch D 1985 Quantum theory, the Church-Turing principle and the universal quantum computer Proc. R. Soc. Lond. A 400 97

[3] Shor P W 1994 Algorithms for quantum computation: discrete logarithms and factoring Proc. of the XXXV Annual Symposium on Foundations of Computer Science pp 124

[4] Unruh W G 1995 Maintaining coherence in quantum computers Phys. Rev. A 51 992–7

[5] Shannon C E 1948 A mathematical theory of communication The Bell System Technical Journal 27 623

[6] Shor P W 1995 Scheme for reducing decoherence in quantum computer memory Phys. Rev. 52 2493

[7] Steane A M 1996 Simple quantum error correction codes Phys. Rev. Letters A 54 4741
[8] Calderbank A R and Shor P W 1996 Good quantum error-correcting codes exist Phys. Rev. A 54 1098
[9] Gottesman D 1996 Class of quantum error-correcting codes saturating the quantum Hamming bound Phys. Rev. A 54 1862
[10] Kitaev A Y 2003 Fault-tolerant quantum computation by anyons Ann. Phys., NY 303 2–30
[11] Bombin H 2013 Topological codes Quantum Error Correction ed D A Lidar and T A Brun (Cambridge: Cambridge University Press)
[12] Bombin H and Martin-Delgado M A 2007 Topological quantum error correction with optimal encoding rate Phys. Rev. 73 062303
[13] Albuquerque C D, Junior R P and da Silva E B 2009 On toric quantum codes Int. J. Pure Appl. Math 50 221
[14] de Albuquerque C D, Palazzo R Jr and da Silva E B 2009 Topological quantum codes on compact surfaces with genus g \geq 2 J. Math. Phys. 50 023513
[15] de Albuquerque C D, Palazzo R Jr and da Silva E B 2010 New classes of topological quantum codes associated with self-dual, quasi self-dual and denser tessellations Quantum Information & Computation 10 0956–70
[16] Delfosse N 2013 Tradeoffs for reliable quantum information storage insurface codes and color codes IEEE International Symposium on Information Theory p 917
[17] Breuckmann N P and Terhal B M 2016 Constructions and noise threshold of hyperbolic surface codes IEEE Trans. Inf. Theory 62 3731
[18] Soares Junior W S and Silva E B 2018 Hyperbolic quantum color codes Quantum Information & Computation 18 308–20
[19] Bombin H and Martin-Delgado M A 2006 Topological quantum distillation Phys. Rev. Lett. 97 180501
[20] Gottesman D and Chuang I L 1999 Quantum teleportation is a universal computational primitive Nature 402 390
[21] Yoder T J and Kim I H 2017 The surface code with a twist Quantum 1 2
[22] Brown B J, Laubscher K, Kesselring M S and Wootters J R 2017 Poking holes and cutting corners to achieve clifford gates with the surface code Phys. Rev. X 7 021029
[23] Nigg D, Mueller M, Martinez E A, Schindler P, Henrich M, Monz T, Martin-Delgado M A and Blatt R 2014 Quantum Computations on a topologically encoded qubit Science 345 302
[24] Kubica A, Yoshida B and Pastawski F 2015 Unfolding the color code New J. Phys. 17 083026+
[25] Anderson J T 2013 Homological stabilizer codes Ann. Phys., NY 330 1–22
[26] Bravyi S B and Kitaev A Y 1998 Quantum codes on a lattice with boundary arXiv:quant-ph/9811052
[27] Cesare C A 2008 Topological code architectures for quantum computation PhD Thesis The University of New Mexico
[28] Delfosse N, Iyer P and Poulin D 2016 Generalized surface codes and packing of logical qubits arXiv:1606.07116
[29] Freedman M H and Meyer D A 2001 Projective plane and planar quantum codes Foundations of Computational Mathematics 1 325–32
[30] Bredon G E 1993 Topology and Geometry (Berlin: Springer)
[31] Vick J W 2012 Homology theory: an introduction to algebraic topology Graduate Texts in Mathematics (New York: Springer)
[32] Giblin P 2010 Graphs, Surfaces and Homology (Cambridge: Cambridge University Press)
[33] Lidar D and Brun T 2013 Quantum Error Correction (Cambridge: Cambridge University Press)