Extended Graph Rotation Systems as a Model for Cyclic Weaving on Orientable Surfaces

ERGUN AKLEMAN, JIANER CHEN
Texas A&M University, College Station, Texas

JONATHAN L. GROSS
Columbia University, New York

Abstract
We present an extension of the theory of graph rotation systems, which has been a widely used model for graph imbeddings on topological surfaces. The extended model is quite beyond what is needed to specify graph imbeddings on surfaces, and it can be used to represent and generate link structures immersed on surfaces. Since link structures immersed on surfaces can be viewed as woven images in 3D space, the extended graph rotation systems provide a well-formulated mathematical model for developing a topologically robust graphics system with strong interactive operations for the design of woven images in 3D mesh-modeling and computer-aided sculpting.

1 Introduction

Some of our recent research [AkCh99, AkChSr03, AkChGr09] in computer graphics reveals how classical topological graph theory [GrTu87, MoTh01, Wh01], especially the theory of graph rotation systems, can be used as a mathematical foundation in the development of a general paradigm for 3D mesh-modeling systems and computer-aided sculpting. In the current paper, we propose a concept of extended graph rotation systems (abbr. EGRS) and study its mathematical properties. Figure 1 shows some examples of the images of cyclic plain-woven objects that were designed and generated using our graphics software developed based on the extended graph rotation systems.

![Venus Bunny Rocker Arm Genus-3 Object](image)

Figure 1: Examples of woven objects constructed from orientable manifold meshes.
Extended graph rotation systems are a generalized version of the standard graph rotation systems, with a generalized edge-twist operation. Most importantly, instead of regarding a graph rotation system as a combinatorial representation of the graph imbedded on a 2-dimensional manifold (i.e., a surface), we interpret the face boundary walks of the imbedding induced by an extended graph rotation system as a topological link (i.e., a collection of inter-linked knots [Ad04]) immersed on a surface. We show in this paper how the extended graph rotation systems are conveniently and effectively used for generating, representing, and operating on link structures immersed on surfaces. When viewing link structures immersed on surfaces as woven structures in 3D-space, the theory of extended graph rotation systems provides a solid mathematical foundation, and it offers powerful techniques for the development of an interactive-graphics system for the design of a variety of 3D woven images.

Adhering to the formal mathematical model presented herein assures topological robustness, which means that the objects generated by our system are always valid link immersions on surfaces, and completeness, which means that it can generate link immersions on all possible surfaces.

From the perspective of graphic artists, the EGRS model underlying the 3D-modeling software system provides a systematic method for dealing with 3D-shapes of complicated topological structures, and it overcomes the problem of large gaps in the weaving structures on surfaces of shapes like those shown in Figure 1. The level of careful detail in Sections 2, 3, and 4 is intended to make implementation of a computer graphics system based on rotation systems reasonably accessible to those beyond the topological graph theory community. In particular, Section 2 provides the prerequisite general background in topological graph theory. As edge-twisting, edge-insertion, and edge-deletion are fundamental surgery operations in graph imbeddings, and are critical to our study on the EGRS model, we re-examine these operations in Sections 3 and 4. More specifically, for each of the operations, we identify certain subtle cases that seem not to have been explicated heretofore in the literature of topological graph theory. We propose algorithms to handle these subtle cases and prove the correctness of our algorithms (see Theorem 3.4, Case (B2), Theorem 4.1, Case (B2), and Theorem 4.2, Case (B2)).

Of course, we cannot provide a mathematical guide to the artistic inspiration involved in the creation of the woven objects in Figure 1. However, from the perspective of a topologist, a graphics software system based on the EGRS model, such as TOPMOD [ACSMT08], enables an artist

1. to imbed a 1-skeleton for the surface underlying the woven image into 3-space, by specifying the coordinates of each vertex, and representing the edges as straight-line segments;
2. to specify the twisting on every edge;
3. and to specify the colors of the threads for the edges.

without needing to understand the underlying mathematics that we present here.

Section 5 gives a concise definition of the EGRS model. Section 6 describes the mathematics that provides a formal description how the graph rotation system model for specifying graph imbeddings, is extended to and re-interpreted as a robust model for specifying link structures immersed on orientable surfaces, which thus induce woven images in 3D-space. After a brief look at knot theory, we show by Theorem 6.3, our main theorem how the EGRS model represents and operates on a cyclic plain-weaving structure, for which all of the previously presented theory, in particular that for the edge surgery operations, becomes a necessity.
For general background on the mathematics of weaving, we recommend the classic paper of Grunbaum and Shephard [GrSh80], which confines its attention to weaving in the plane.

2 Preliminaries on Topological Graph Theory

Rather than presenting any new results in topological graph theory, this section confines itself to some necessary background. It is consistent with the discussions of these issues to be found in [GrTu87, BWGT09]. Alternative perspectives are provided by [MoTh01, Wh01].

2.1 Graphs and surfaces

Our graphs are always undirected. Multiple edges with the same endpoints and self-loops are allowed. A self-loop has only one endpoint, yet it has two distinguishable edge-ends. The distinction between the two edge-ends of a self-loop is achieved topologically by regarding the interior of each edge as parametrized by the open unit interval (0, 1). The edge-ends are small neighborhoods of the limit points 0 and 1, respectively, and, thus, they are distinguishable, even for a self-loop. This permits us to differentiate, for instance, between the two possible directions in which one can traverse a self-loop.

Any edge can be specified by a pair \([v, w]\) of vertices, almost as one might do when the discussion is confined to simple graphs. To adapt this form of specification to a multi-edge of multiplicity \(m\), we can use \(m\) different names for one of their common endpoints and another \(m\) different names for the other endpoint. This satisfies our need for distinct edges within a multi-edge to have distinguishable names. When an edge is a self-loop, two names for the same endpoint \(v\) and \(w\) may be interpreted, when context requires, as the two distinguishable edge-ends of the edge. Under these conventions, each edge \(e = [v, w]\) induces two oriented edges \(\langle v, w \rangle\) and \(\langle w, v \rangle\), each running from one edge-end of edge \(e\) to the other edge-end of \(e\). We may sometimes denote the two oriented edges arising from \(e\) by \(e^+\) and \(e^-\).

A surface is a compact boundaryless 2-dimensional manifold, which can be either orientable or non-orientable [Ch90, ChGrRi94, GrTu87]. An imbedding of a graph \(G\) on a surface \(S\) is a homeomorphism of the graph onto a topological subspace of the surface. It is always assumed to be cellular, which means that every connected component of \(S - G\), i.e., the interior of each face in the imbedding is homeomorphic to an open disk. Therefore, the surface \(S\) can be reconstructed, uniquely, from a graph \(G\) imbedded on the surface if the boundaries of all faces in the imbedding are specified — it is achieved simply by pasting the perimeter of a proper polygon along each face boundary in the imbedding. In fact, it has been a common practice both in topological graph theory [GrTu87], as we describe in §2.2 and in computer graphics [Ba72] to represent surfaces along with graph imbeddings.\(^1\)

\(^1\)We remark that we have adopted the concept of “oriented edges” from the literature of standard topological graph theory. It should be noted that our graphs are undirected so edges in a graph have no directions. However, when the face boundaries of a graph imbedding on a surface is traversed, it is critical to specify the order of the two ends of an edge in which the traversing proceeds, which involves the concept of oriented edges.

\(^2\)In computer graphics, graph imbeddings on surfaces are called “meshes”, and surface modeling based on graph imbeddings is called “mesh-modeling” [AkChSr03, ACSMT08, Ma88].
2.2 Specifying imbeddings by rotation systems

Graph rotation systems are used to give a purely combinatorial specification of an imbedding of a graph in a surface.

**Definition** Let \( G \) be a graph with \( n \) vertices. A rotation at a vertex \( v \) of \( G \) is a cyclic ordering of the oriented edges originating at \( v \). A (pure) rotation system of the graph \( G \) consists of a set of \( n \) rotations, one for each vertex of \( G \).

**Remark** The spatial imagery in conceptualizing a rotation at a vertex \( v \) is that for each edge-end at \( v \), there is an oriented edge that begins at that edge-end. Thus, a self-loop is associated with two oriented edges in the rotation at \( v \), representing the two respective directions in which the self-loop could be traversed.

It is easy to see that an imbedding \( \pi(G) \) of a graph \( G \) on an oriented surface naturally induces a pure rotation system of the graph \( G \). Also, note that the face-boundary walk (for short, \( fb \)-walk) of each face in \( \pi(G) \) is a closed walk, that is, a (cyclically ordered) sequence of oriented edges. Conversely, it has been well-known since [He1891] and [Ed60] (for simple graphs) and [GrAl74] (for general graphs) that a pure rotation system \( \rho(G) \) of a graph \( G \) uniquely determines an imbedding of \( G \) on an oriented surface \( S \), and, thus, uniquely determines the surface \( S \). To do this, we need a face-tracing algorithm (see Algorithm 1, page 6. Note that for the pure rotation system \( \rho(G) \), all edges have twist-type 0). The face-tracing algorithm will uniquely construct all \( fb \)-walks in \( \rho(G) \), from which we can match each \( fb \)-walk of length \( s \) with the perimeter of an \( s \)-sided polygon. This process, as explained above, will uniquely reconstruct the surface \( S \). A full exposition of rotation systems and face-tracing, with examples, is given in §3.2 of [GrTu87].

**Definition** Given a rotation system \( \rho(G) \) for a graph \( G \), a face corner (sometimes just corner) is a triple \((v, e, e')\), comprising a vertex \( v \) and two oriented edges \( e = \langle v, u \rangle \) and \( e' = \langle v, u' \rangle \), both oriented out of \( v \), where the \( v \)-edge-end of \( e' \) immediately follows the \( v \)-edge-end of \( e \) in the rotation at \( v \), as illustrated in Figure 2. If neither \( e \) nor \( e' \) is a self-loop, we say that \( e' \) is the 0-next to \( e \) at \( v \) and that \( e \) is the 1-next to \( e' \) at \( v \). For a self-loop, we must say which orientation is 0-next or 1-next.

![Figure 2: The corner \((v, e, e')\), where \( e = vu \) and \( e' = vu' \).](image)

**Remark** We emphasize that in this convention for specifying a corner, both of the oriented edges point out from \( v \). When we write \((v, e, e')\), it means that \( e' \) follows \( e \) in the cyclic order at vertex \( v \). In the special case of a vertex \( v \) of degree 2, there are corners \((v, e, e')\) and \((v, e', e)\).

2.3 Rotation systems and surgery

Surgery operations on pure graph rotation systems (thus, on graph imbeddings in orientable surfaces) have been extensively studied [Ch90, GrTu87]. For example, inserting an edge into a pure graph rotation system adheres to the following rules:
Extended Graph Rotation Systems as a Model for Cyclic Weaving

**Edge-Insert-0**

- If the two ends of an edge are inserted into two corners of the same face, then the edge splits that face into two faces, and the two oriented edges corresponding to the new edge belong to the fb-walks of those two different faces in the new imbedding. Figure 3 illustrates the splitting of a face with boundary walk $abcef$ into two faces, with respective boundary walks $abcd^-$ and $d^+ef$.

![Figure 3: Inserting the edge $d$ into the face with boundary walk $abcef$ yields the faces $abcd^-$ and $d^+ef$.](image)

- If the two ends of an edge are inserted between two corners of two different faces, then the inserted edge merges those two faces into a single face, and the two oriented edges corresponding to the new edge belong to the fb-walk of that single face in the new imbedding. Figure 4 illustrates the merger of two faces with respective boundary walks $abc$ and $efgh$ into a single face, with boundary walk $abcd^+ - efghd^-$.

![Figure 4: Inserting the respective ends of edge $d$ into two different faces merges them into a single face.](image)

The operation of edge-deletion on a pure rotation system “reverses” edge insertion.

**Edge-Delete-0**

- If the two oriented edges corresponding to an edge $e$ appear in the fb-walks of two different faces, then deleting the edge $e$ merges the two faces into a single face. This is illustrated in Figure 5.

![Figure 5: When edge $d$ appears in two different fb-walks, its deletion merges the rest of the two fb-walks (i.e., excluding $d$) into the fb-walk of a single face.](image)

- If the two oriented edges corresponding to an edge $e$ belong to the fb-walk of a single face, then deleting the edge $e$ splits that face into two faces. This is illustrated in Figure 6. 

![Figure 6:](image)
**Definition** A graph rotation system can be augmented to represent an imbedding on a non-orientable surface. For this, we label each edge \([u,v]\) in the graph either as type-0 (flat) or type-1 (twisted), which we call the twist-type of \([u,v]\) and denote by \(tw-type([u,v])\).

**Definition** A general rotation system of a graph \(G = (V,E)\) consists of a pure rotation system of \(G\) plus a mapping \(tw-type : E \rightarrow \{0,1\}\) that assigns to each edge of \(G\) a twist-type.

It is also known that an imbedding of a graph \(G\) on any surface (orientable or non-orientable) induces a general rotation system for the graph \(G\), and that each general rotation system for the graph \(G\) induces an imbedding of \(G\) on an (orientable or non-orientable) surface. Moreover, the face-tracing algorithm can be generalized [GrTu87] to apply to general graph rotation systems. For convenience in our later discussion, we now present the **General Face-Tracing Algorithm**, which is a modification of the algorithm described in [GrTu87], as Algorithm 1. A version of Algorithm 1 appears in [AkChGr09].

**Algorithm 1** General Face-Tracing Algorithm.

**Subroutine** FaceTrace\((t_0, \langle u_0, w_0 \rangle)\)

*Input:* an oriented edge \(\langle u_0, w_0 \rangle\); and a number \(t_0 \in \{0,1\}\), called its trace-type.

*Remark:* the trace-type corresponds to direction of traversal of the edge.

*Output:* a sequence of oriented edges that forms an fb-walk containing \(\langle u_0, w_0 \rangle\).

1. trace and print \(\langle u_0, w_0 \rangle\);
2. \(t = t_0 + tw-type([u_0, w_0]) \mod 2\);
3. \(\langle u, w \rangle = \text{the } t\text{-next to }\langle w_0, u_0 \rangle \text{ at } w_0; \ \ \ \ \ \langle u = w_0\)
4. **while** \((\langle u, w \rangle \neq \langle u_0, w_0 \rangle) \text{ or } (t \neq t_0)\) **do**
   - trace and print \(\langle u, w \rangle\);
   - \(t = t + tw-type([u, w]) \mod 2\);
   - \(\langle w', u' \rangle = \langle w, u \rangle\);
   - \(\langle u, w \rangle = \text{the } t\text{-next to }\langle w', u' \rangle \text{ at } w'. \ \ \ \ \ \langle u = w'\)

**Algorithm** FbWalks\((\rho(G))\)

*Input:* \(\rho(G)\) is a general graph rotation system.

*Output:* the collection of all fb-walks in \(\rho(G)\).

**while** there is an untraced face corner \(\langle u, e, e' \rangle\) in \(\rho(G)\) **do**
   - suppose that \(e' = \langle u, w \rangle\); call FaceTrace\((0, \langle u, w \rangle)\).

**Remark** We observe that whereas face-tracing on pure rotation systems can stop when all the oriented edges have been traced, face-tracing stops on general rotation systems when all the face corners have been traced.
3 On the Edge-Twisting Operation

Surgical edge operations have been powerful tools in the study of graph imbeddings in topological graph theory [GrTu87, MoTh01, Wh01] and in the research in computer graphics and the development of graphics softwares [GuSt85, Ma88]. In particular, these operations offer the foundation for the development of a graphics mesh modeling system with a highly interactive user-interface [AkChSr00, AkChSr03, ACSMT08]. Surgical edge operations on pure graph rotation systems have been extensively studied and well understood. However, when edge-twists are introduced, certain fairly subtle phenomena occur. In fact, some of these intricacies seem to have not been addressed carefully in the literature of either topological graph theory or computer graphics.

The underlying theory for our EGRS model is heavily based on edge-twists, and our graphics modeling system includes a highly interactive user-interface based on EGRS, whose development and analysis require a deep and thorough understanding of the objects and the surgical edge operations included within the EGRS model. In particular, the EGRS model for weaving regards an edge in the model as a thin strip, in which the sides of the strip are regarded as two threads in the weave. Twisting an edge in the EGRS model specifies that these two threads wind around each other (see Section 6 for more details). This is quite different from the meaning of twisting an edge of a general rotation system in pure topological graph theory.

We proceed with a detailed, formal study of surgical edge operations on general graph rotation systems, from which we can easily extend to the EGRS model. This section is focused on the edge-twisting operation, and Section 4 is devoted to the edge-insertion and edge-deletion operations. In particular, Theorem 3.4, Theorem 4.1, and Theorem 4.2 are new theoretical results that repair imprecisions in what has been described elsewhere.

Our terminology and notations here are a bit different from what may be found elsewhere. Throughout the discussion of Sections 3-4, we assume that $\rho(G)$ is a general graph rotation system, and we consider the effect of applying an edge-operation to an edge $e$ in the graph $G$.

**Definition** The operation of **twisting an edge $e$ in a general rotation system $\rho(G)$** means changing its type, either from 0 to 1, or from 1 to 0. (When traversing a twisted edge during face-tracing, the cyclic direction at the terminating vertex at which one selects the next oriented edge is taken to be opposite from the direction at the originating vertex.)

### 3.1 Calculating an fb-walk

We closely examine the subroutine FaceTrace, as given within Algorithm 1.

**Definition** Let $t$ be the current trace-type and $(u, w)$ the next oriented edge to be traced, at some moment during the face-tracing process. The pair $(t, (u, w))$ is called a **trace-pair**.

Note that the current trace-pair $(t, (u, w))$ uniquely determines the next trace-type $t'$, by the rule $t' = t + \text{type}((u, w)) \pmod{2}$; and then the trace-type $t'$ combined with the oriented edge $(u, w)$ uniquely determines the next oriented edge $(w, v)$ to be traced: that is, $(w, v)$ is $t'$-next to $(w, u)$ at $w$. Therefore, any trace-pair $(t, (u, w))$ that occurs during the face-tracing process completely determines the fb-walk that is constructed by calling FaceTrace$(t, (u, w))$. 

Lemma 3.1 Let \( h \geq 1 \). Suppose that in a given general rotation system \( \rho(G) \) the first \( h \) trace-pairs traced by calling \( \text{FaceTrace}(t_0, \langle u_0, u_1 \rangle) \) are

\[
( t_0, \langle u_0, u_1 \rangle ), \ ( t_1, \langle u_1, u_2 \rangle ), \ldots, \ ( t_{h-1}, \langle u_{h-1}, u_h \rangle ),
\]

and suppose that after tracing the \( h^{th} \) oriented edge \( \langle u_{h-1}, u_h \rangle \), the process has trace-type \( t_h \). Then the first \( h \) trace-pairs traced by \( \text{FaceTrace}(t_h, \langle u_h, u_{h-1} \rangle) \), where \( t'_h = t_h + 1 \mod 2 \), are

\[
( t'_1, \langle u_h, u_{h-1} \rangle ), \ldots, \ ( t'_2, \langle u_2, u_1 \rangle ), \ ( t'_1, \langle u_1, u_0 \rangle ),
\]

where \( t'_i = t_i + 1 \mod 2 \) for all \( i, h \geq i \geq 1 \), and after tracing the \( h^{th} \) oriented edge \( \langle u_1, u_0 \rangle \), the process has current trace-type \( t'_0 = t_0 + 1 \mod 2 \).

Proof. The lemma can be easily verified for the case \( h = 1 \). We proceed by induction on \( h \).

Suppose that \( h > 1 \). For each \( i \) such that \( 0 \leq i \leq h \), let

\[
\tau_i = \text{tw-type}(\langle u_0, u_1 \rangle) + \cdots + \text{tw-type}(\langle u_{i-1}, u_i \rangle).
\]

After tracing the \( i^{th} \) oriented edge \( \langle u_{i-1}, u_i \rangle \), the process \( \text{FaceTrace}(t_0, \langle u_0, u_1 \rangle) \) has trace-type \( t_i = t_0 + \tau_i \mod 2 \). Thus, \( t_h = t_0 + \tau_h \mod 2 \), which gives \( t'_h = t_h + 1 + \tau_h \mod 2 \). Now consider the process \( \text{FaceTrace}(t'_h, \langle u_h, u_{h-1} \rangle) \). After tracing the first oriented edge \( \langle u_h, u_{h-1} \rangle \), the trace-type \( t'_{h-1} \) of the process becomes

\[
t'_{h-1} + \text{tw-type}(\langle u_{h-1}, u_h \rangle) = t_0 + 1 + \tau_{h-1} = t_h + 1 \mod 2.
\]

This proves that \( t'_{h-1} = t_h + 1 \mod 2 \).

By the premises of this lemma, the tracing process \( \text{FaceTrace}(t_0, \langle u_0, u_1 \rangle) \) has trace-type \( t_{h-1} \) after tracing \( \langle u_{h-2}, u_{h-1} \rangle \), and the oriented edge \( \langle u_{h-1}, u_h \rangle \) follows the oriented edge \( \langle u_{h-2}, u_{h-1} \rangle \) in that process. Thus, the oriented edge \( \langle u_{h-1}, u_h \rangle \) must be \( t_h \)-next to the oriented edge \( \langle u_{h-2}, u_{h-1} \rangle \) at \( u_{h-1} \). In consequence, the edge \( \langle u_{h-1}, u_{h-2} \rangle \) must be \( t'_h \)-next to \( \langle u_{h-1}, u_h \rangle \) at \( u_{h-1} \). Now since the process \( \text{FaceTrace}(t'_h, \langle u_h, u_{h-1} \rangle) \) has a trace-type \( t'_{h-1} \) after tracing the first oriented edge \( \langle u_h, u_{h-1} \rangle \), the next oriented edge to be traced by the process must be \( \langle u_{h-1}, u_{h-2} \rangle \). Therefore, after tracing the first oriented edge \( \langle u_h, u_{h-1} \rangle \), the process \( \text{FaceTrace}(t'_h, \langle u_h, u_{h-1} \rangle) \) has a trace-pair \( \langle t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle \rangle \), and accordingly, it will copy the process \( \text{FaceTrace}(t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle) \) when tracing the next \( h-1 \) trace-pairs. By the induction hypothesis, the first \( h-1 \) trace-pairs by the subroutine \( \text{FaceTrace}(t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle) \) are

\[
( t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle ), \ldots, \ ( t'_2, \langle u_2, u_1 \rangle ), \ ( t'_1, \langle u_1, u_0 \rangle ),
\]

where \( t'_i = t_i + 1 \mod 2 \) for all \( i \) such that \( h-1 \geq i \geq 1 \), and after tracing the \( h-1 \)th oriented edge \( \langle u_i, u_0 \rangle \), the process has trace-type \( t'_0 = t_0 + 1 \mod 2 \). Combining this with the first traced oriented edge \( \langle u_h, u_{h-1} \rangle \), we conclude that the first \( h \) trace-pairs traced by the subroutine \( \text{FaceTrace}(t'_h, \langle u_h, u_{h-1} \rangle) \) are

\[
( t'_h, \langle u_h, u_{h-1} \rangle ), \ldots, \ ( t'_2, \langle u_2, u_1 \rangle ), \ ( t'_1, \langle u_1, u_0 \rangle ),
\]

where \( t'_i = t_i + 1 \mod 2 \) for all \( i \) with \( h \geq i \geq 1 \), and that after tracing the \( h^{th} \) oriented edge \( \langle u_1, u_0 \rangle \), the process has trace-type \( t_0 + 1 \mod 2 \).
3.2 Reversing an fb-walk

**Definition** Suppose that the fb-walk \( B \) of a face \( F \) is the cyclically ordered sequence of trace-pairs

\[
(0, \langle u_0, u_1 \rangle), (1, \langle u_1, u_2 \rangle), \ldots, (t_h-2, \langle u_{h-2}, u_{h-1} \rangle), (t_{h-1}, \langle u_{h-1}, u_0 \rangle).
\]

Then the **reverse of the fb-walk** \( B \) is defined to be the cyclically ordered sequence of trace-pairs

\[
B^r = (t'_0, \langle u_0, u_{h-1} \rangle), (t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle) , \ldots, (t'_2, \langle u_2, u_1 \rangle), (t'_1, \langle u_1, u_0 \rangle),
\]

where \( t'_i = t_i + 1 \mod 2 \) for all \( i \) such that \( h - 1 \geq i \geq 0 \).

**Corollary 3.2** Let \( (u_0, \langle u_0, u_{h-1} \rangle, \langle u_0, u_1 \rangle) \) be a face corner induced by a fixed general rotation system \( \rho(G) \) for a graph \( G \). If the subroutine call \( \text{FaceTrace}(0, \langle u_0, u_1 \rangle) \) traces an fb-walk \( B \), then the fb-walk traversed by the call \( \text{FaceTrace}(1, \langle u_0, u_{h-1} \rangle) \) is the reverse of \( B \).

**Proof.** First note that the last oriented edge on the fb-walk \( B \) traced by the subroutine call \( \text{FaceTrace}(0, \langle u_0, u_1 \rangle) \) must be \( \langle u_{h-1}, u_0 \rangle \), and that after tracing \( \langle u_{h-1}, u_0 \rangle \) the process must have trace-type 0 — otherwise, the process would not have stopped. Therefore, we can assume that the fb-walk \( B \) is composed of the cyclically ordered trace-pairs

\[
(0, \langle u_0, u_1 \rangle), (1, \langle u_1, u_2 \rangle), \ldots, (t_{h-1}, \langle u_{h-2}, u_{h-1} \rangle), (t_{h-1}, \langle u_{h-1}, u_0 \rangle).
\]

By Lemma 3.1, the first \( h \) trace-pairs by the subroutine \( \text{FaceTrace}(1, \langle u_0, u_{h-1} \rangle) \) are

\[
(1, \langle u_0, u_{h-1} \rangle), (t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle), \ldots, (t'_2, \langle u_2, u_1 \rangle), (t'_1, \langle u_1, u_0 \rangle),
\]

where \( t'_i = t_i + 1 \mod 2 \), for all \( i \) such that \( h - 1 \geq i \geq 1 \), and after tracing \( \langle u_1, u_0 \rangle \), the trace-type is 1. Since \( \langle 0, u_{h-1} \rangle \) is 1-next to \( \langle u_0, u_1 \rangle \) at \( u_0 \), the next edge to be traced by \( \text{FaceTrace}(1, \langle u_0, u_{h-1} \rangle) \) is \( \langle u_0, u_{h-1} \rangle \), so the face-tracing process terminates. Therefore, the sequence in Eq (3.1) gives the entire fb-walk traced by the process \( \text{FaceTrace}(1, \langle u_0, u_{h-1} \rangle) \), which is the reverse of \( B \). □

Since the fb-walk \( B \) and its reverse bound the same topological region in the imbedding induced by the rotation system \( \rho(G) \), only one of them is needed. Therefore, the face corner \( (u_0, \langle u_0, u_{h-1} \rangle, \langle u_0, u_1 \rangle) \) is regarded as “traced” if the corner is traced either in the order \( \langle u_{h-1}, u_0 \rangle, u_0, (u_0, u_1) \), or in the order \( \langle u_1, u_0 \rangle, u_0, \langle u_0, u_{h-1} \rangle \).

**Corollary 3.3** Fix a general rotation system \( \rho(G) \) and let \( \langle u_0, u_1 \rangle \) be an edge of \( G \). Suppose that \( \text{FaceTrace}(t, \langle u_0, u_1 \rangle) \) traces an fb-walk \( B \). Then the fb-walk traced by calling \( \text{FaceTrace}(t', \langle u_1, u_0 \rangle) \), where \( t' = t + 1 + \text{type}([u_0, u_1]) \mod 2 \), is the reverse walk \( B^r \).

**Proof.** First suppose that \( t = 0 \). Consider the face corner \( (u_0, \langle u_0, u_{h-1} \rangle, \langle u_0, u_1 \rangle) \) in \( \rho(G) \). As discussed in the proof of Corollary 3.2, the fb-walk \( B \) traced by \( \text{FaceTrace}(0, \langle u_0, u_1 \rangle) \) can be assumed to be

\[
(0, \langle u_0, u_1 \rangle), (1, \langle u_1, u_2 \rangle), \ldots, (t_{h-2}, \langle u_{h-2}, u_{h-1} \rangle), (t_{h-1}, \langle u_{h-1}, u_0 \rangle).
\]

Using Corollary 3.2, we infer that the fb-walk traced by \( \text{FaceTrace}(1, \langle u_0, u_{h-1} \rangle) \) is

\[
(1, \langle u_0, u_{h-1} \rangle), (t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle), \ldots, (t'_2, \langle u_2, u_1 \rangle), (t'_1, \langle u_1, u_0 \rangle),
\]

which we observe is \( B^r \), the reverse of \( B \). Immediately before tracing \( \langle u_1, u_0 \rangle \), the process \( \text{FaceTrace}(1, \langle u_0, u_{h-1} \rangle) \) has trace-type

\[
t'_1 = 1 + \text{type}([u_0, u_{h-1}]) + \cdots + \text{type}([u_2, u_1]) = 1 + \text{type}([u_1, u_0]) \mod 2.
\]
Therefore, the fb-walk $B^r$ is also produced by the process FaceTrace($t'_1, \langle u_1, u_0 \rangle$), where

$$t'_1 = 1 + \text{type([u_0, u_1])} = t + 1 + \text{type([u_0, u_1])} = t' \pmod{2}$$

This proves that FaceTrace($t'_1, \langle u_1, u_0 \rangle$) also traces the reverse $B^r$ of the fb-walk $B$.

Alternatively, suppose that $t = 1$. In this case the face corner to be considered is $(u_0, \langle u_0, u_1 \rangle, \langle u_0, u_{h-1} \rangle)$. Here, too, it can be easily seen that the fb-walk $B$ traced by FaceTrace(1, $\langle u_0, u_1 \rangle$) can be assumed to be of the form

$$(1, \langle u_0, u_1 \rangle), (t_1, \langle u_1, u_2 \rangle), \ldots, (t_{h-2}, \langle u_{h-2}, u_{h-1} \rangle), (t_{h-1}, \langle u_{h-1}, u_0 \rangle).$$

Corollary 3.2 implies that the fb-walk traced by FaceTrace(0, $\langle u_0, u_{h-1} \rangle$) is

$$(0, \langle u_0, u_{h-1} \rangle), (t'_{h-1}, \langle u_{h-1}, u_{h-2} \rangle), \ldots, (t'_2, \langle u_2, u_1 \rangle), (t'_1, \langle u_1, u_0 \rangle).$$  \hspace{1cm} (3.2)

and that FaceTrace(0, $\langle u_0, u_{h-1} \rangle$) traces the reverse $B^r$ of the fb-walk $B$. (Note that the reverse of the reverse of a walk $B$ is the walk $B$ itself). Therefore, immediately before the process FaceTrace(0, $\langle u_0, u_{h-1} \rangle$) traces the last oriented edge $\langle u_1, u_0 \rangle$ in the trace-pair sequence (3.2), the trace-type is

$$t'_1 = \text{type([u_0, u_{h-1}])} + \cdots + \text{type([u_2, u_1])} \pmod{2}$$

$$= \text{type([u_0, u_1])}$$

$$= t + 1 + \text{type([u_0, u_1])} \pmod{2}$$

$$= t'.$$

Therefore, the process FaceTrace($t'_1, \langle u_1, u_0 \rangle$) and the process FaceTrace(0, $\langle u_0, u_{h-1} \rangle$) trace the same fb-walk $B^r$, which is the reverse of $B$. \hfill \square

Accordingly, for each oriented edge $\langle u, w \rangle$, there are only two essentially different trace-pairs. By Corollary 3.3, the four trace-pairs $(0, \langle u, w \rangle), (1, \langle u, w \rangle), (0, \langle w, u \rangle)$ and $(1, \langle w, u \rangle)$ of an edge $[u, w]$ correspond to only two different trace-pairs used in the face tracing process Facing($\rho(G)$). These trace-pairs will be called the trace-pairs induced on the edge $[u, w]$. The trace-pairs $(0, \langle u, w \rangle), (1, \langle u, w \rangle)$ are said to use the oriented edge $\langle u, w \rangle$, and the trace-pairs $(0, \langle w, u \rangle)$ and $(1, \langle w, u \rangle)$ are said to use the oriented edge $\langle w, u \rangle$.

### 3.3 Effects of edge-twisting surgery

**Theorem 3.4** Twisting an edge $e$ in a general rotation system $\rho(G)$ satisfies the following rules:

- (A) Suppose that the two trace-pairs induced by edge $e$ belong to the fb-walks of two different faces in the imbedding. Then twisting edge $e$ merges the two faces into a single face;

- (B) Suppose that the two trace-pairs induced by edge $e$ belong to the fb-walk of the same face $F$ in the imbedding.

  - (B1) If the two trace-pairs induced by $e$ use the same oriented edge, then twisting edge $e$ splits the face $F$ into two faces;

  - (B2) If the two trace-pairs induced by edge $e$ use different oriented edges, then twisting edge $e$ converts the face $F$ into a new single face.
Now we twist the edge $e = [u, w]$. We first consider case (A). By Corollary 3.3, we can assume that the two faces whose fb-walks contain the trace-pairs induced by $e$ are

$$F_0 = \text{FaceTrace}(0, \langle u, w \rangle) : (0, \langle u, w \rangle), (t_0, \langle w, w_0 \rangle), \ldots, (t'_0, \langle x_0, u \rangle),$$

and

$$F_1 = \text{FaceTrace}(1, \langle u, w \rangle) : (1, \langle u, w \rangle), (t_1, \langle w, w_1 \rangle), \ldots, (t'_1, \langle x_1, u \rangle),$$

where we have $t_0 + t_1 = 1$. Note that no trace-pairs induced by the edge $[u, w]$ can appear anywhere else except the two places given in the above sequences. See the top part of Figure 7(1) for an illustration.

![Figure 7: Twisting an edge in a general rotation system](image)

Now we twist the edge $e$ (i.e., changing the twist-type, as shown at the bottom of Figure 7(1)), and we thereby obtain a new rotation system $\rho'(G)$ of the graph $G$. We apply $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ to $\rho'(G)$. Since no vertex rotation is changed and no other edge changes the twist-type, $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ on $\rho'(G)$ follows the same sequence of fb-walk $F_0$ (starting from $(t_0, \langle w, w_0 \rangle)$), until it traces the trace-pair $(0, \langle u, w \rangle)$. Now since the edge $e = [u, w]$ changes twist-type, after tracing $(0, \langle u, w \rangle)$, the trace-type becomes $t_0 + 1 = t_1 \mod 2$. From the sequence for $F_1$, we know that $\langle w, w_1 \rangle$ is the $t_1$-next to $\langle w, u \rangle$ at $w$. Therefore, the next oriented edge to be traced by $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ on $\rho'(G)$ should be $\langle w, w_1 \rangle$ (see the bottom of Figure 7(1)). Thus, after tracing the trace-pair $(0, \langle u, w \rangle)$, the process $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ on $\rho'(G)$ has a trace-pair $(t_1, \langle w, w_1 \rangle)$. Again since no vertex rotation is changed and no other edge changes the twist-type, $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ now will follow the sequence $F_1$ (starting from $(t_1, \langle w, w_1 \rangle)$) until it traces the trace-pair $(1, \langle u, w \rangle)$. By an analysis similar to that above, after tracing $(1, \langle u, w \rangle)$, we see that the process $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ has a trace-type $t_0$ and that the next edge to be traced is $\langle w, w_0 \rangle$ (see the bottom of Figure 7(1)). Therefore, the process closes its tracing of the fb-walk.

Since the process $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ on $\rho'(G)$ uses exactly the same trace-pairs used in fb-walks $F_0$ and $F_1$, and since no vertex rotation is changed and no other edge changes twist-type, we conclude that the set of faces for the rotation system $\rho'(G)$ can be obtained from that for $\rho(G)$ by replacing the fb-walks $F_0$ and $F_1$ in $\rho(G)$ by a single fb-walk produced by $\text{FaceTrace}(t_0, \langle w, w_0 \rangle)$ on $\rho'(G)$. This completes the proof for case (A).
We next consider case (B1). By the premise, we may assume that the fb-walk $F$ is
(see the top of Figure 7(2))
$$F : (t_0, \langle w, w_0 \rangle), \ldots, (0, \langle u, w \rangle), (t_1, \langle w, w_1 \rangle), \ldots, (1, \langle u, w \rangle).$$
Note that in the above sequence, we have $t_0 + t_1 = 1$. Twisting the edge $e = [u, w]$ gives a new rotation system $\rho'(G)$ of the graph $G$. We apply FaceTrace($t_0, \langle w, w_0 \rangle$) to $\rho'(G)$. By the same reasoning as that in (A), the process follows the sequence of $F$ (starting from $(t_0, \langle w, w_0 \rangle)$) until it traces the middle trace-pair $(0, \langle u, w \rangle)$. Since the twist-type of edge $[u, w]$ has changed, the trace-type after tracing the edge $(u, w)$ is $t_1 + 1 = t_0 \pmod{2}$. In the sequence of fb-walk $F$, edge $(w, w_0)$ is $t_0$-next to edge $(w, u)$ at $w$. Therefore, the process FaceTrace($t_0, \langle w, w_0 \rangle$) on $\rho'(G)$ closes its tracing with the fb-walk (see the bottom of Figure 7(2))
$$F_0 : (t_0, \langle w, w_0 \rangle), \ldots, (0, \langle u, w \rangle).$$

Completely similarly, we observe that applying FaceTrace($t_1, \langle w, w_1 \rangle$) on $\rho'(G)$ will result in the fb-walk
$$F_1 : (t_1, \langle w, w_1 \rangle), \ldots, (1, \langle u, w \rangle).$$
Since the trace-pairs used by $F_0$ and $F_1$ are exactly those used by $F$, we conclude changing the twist-type of edge $e$ splits the face $F$ into two faces $F_0$ and $F_1$.

Finally we discuss case (B2), which is not mentioned in [GrTu87] or elsewhere. By the premise, we may assume that the fb-walk of $F$ is (see the top of Figure 7(3))
$$F : (0, \langle u, w \rangle), (t_1, \langle w, w_1 \rangle), \ldots, (t, \langle x_2, w \rangle), (t_1, \langle w, u \rangle), (0, \langle u, w_2 \rangle), \ldots, (t', \langle x_1, u \rangle),$$
where the trace-types $t$ and $t'$ are irrelevant, and we have let $t_1 = \text{type}([u, w])$. We define $t_0 = t_1 + 1 \pmod{2}$. Note that the trace-type before the oriented edge $(w, u)$ in the sequence is also $t_1$; by Corollary 3.3, the trace-pairs $(0, \langle u, w \rangle)$ and $(t_0, \langle w, u \rangle)$ are equivalent.

Within the sequence $F$, we observe the following:
- (b1). $\langle u, w \rangle$ is $0$-next to $\langle u, x_1 \rangle$ at $u$;
- (b2). $\langle u, w_2 \rangle$ is $0$-next to $\langle u, w \rangle$ at $u$;
- (b3). $\langle w, x_2 \rangle$ is $t_0$-next to $\langle w, u \rangle$ at $w$;
- (b4). $\langle w, u \rangle$ is $t_0$-next to $\langle w, w_1 \rangle$ at $w$;

where (b3) follows because $(w, u)$ is $t_1$-next to $(w, x_2)$ at $w$, and (b4) because $(w, w_1)$ is $t_1$-next to $(w, u)$ at $w$.

From the subsequence $F' : (t_1, \langle w, w_1 \rangle), \ldots, (t, \langle x_2, w \rangle), t_1 \in F$ (where the last $t_1$ is the trace-type after tracing $F'$), by Lemma 3.1, FaceTrace($t_0, \langle w, x_2 \rangle$) will produce the reverse of $F'$ as $F'' : (t_0, \langle w, x_2 \rangle), \ldots, (t'', \langle w_1, w \rangle), t_0$ (where the trace-type $t''$ is irrelevant).

We twist the edge $[u, w]$ (i.e., changing the twist-type of $[u, w]$ to $t_0$), and we apply FaceTrace($0, \langle u, w \rangle$) to the new rotation system $\rho'(G)$. After tracing $(0, \langle u, w \rangle)$, the trace-type becomes $t_0$. By (b3), the next oriented edge to be traced by the process is $\langle w, x_2 \rangle$. Therefore, the modified process will mimic the process FaceTrace($t_0, \langle w, x_2 \rangle$) on the original rotation system $\rho(G)$ and produce the subsequence
$$F'' : (t_0, \langle w, x_2 \rangle), \ldots, (t'', \langle w_1, w \rangle), t_0,$$
until it returns to the vertex $w$ from $\langle w_1, w \rangle$. Note that after tracing $\langle w_1, w \rangle$, the process has a trace-type $t_0$. Now by (b4), the process continues tracing the trace-pair $(t_0, \langle w, u \rangle)$ and changes its trace-type to $t_0 + t_0 = 0 \pmod{2}$. By (b2), the next
oriented edge to be traced thereafter is $(u, w_2)$. Hence, the process then follows the sequence of fb-walk $F$ (starting from $(0, (u, w_2))$) until it traces the trace-pair $(t', (x_1, u))$ — after tracing $(x_1, u)$, the process has a trace-type 0. By (b1), the next edge to be traced is $(u, w)$. Therefore, the process FaceTrace(0, $(u, w)$) on $\rho'(G)$ produces the fb-walk $F'$, given by

$$(0, (u, w)), (t_0, (w, x_2)), \ldots, (t'', (w_1, w)), (t_0, (w, u)), (0, (u, w_2)), \ldots, (t', (x_1, u)).$$

which uses exactly the same oriented edges as fb-walk $F$. In conclusion, twisting the edge $[u, w]$ in this case replaces the face $F$ by a single new face $F'$.

Theorem 3.4 is consistent with what is described by [St78] and [Zh96]. Results on the set of fb-walks of twisting an edge whose two induced trace-pairs both belong to the same fb-walk, that is, case (B2) of Theorem 3.4, seem less than explicit in the existing literature on topological graph theory.

4 On Edge-Insertion and Edge-Deletion

Edge insertion/deletion operations have been fundamental in the studies of pure graph rotation systems on orientable imbeddings [Ch90, GrTu87] and in their applications to computer graphics [AkCh99]. One might expect that most corresponding results involving insertion and deletion operations on pure graph rotation systems would extend naturally to general graph rotation systems. However, there seem to be certain subtle issues that are quite different, and which, to our knowledge, have not been thoroughly studied in the literature. In particular, the parts of Theorems 4.1 and 4.2 concerned with Case (B2) are new. Accordingly, we present a revised study of those operations on general graph rotation systems in this section.

To observe how the surgery operations of edge-insertion and edge-deletion are changed, we consider the general graph rotation systems whose rotation projections [GrTu87] are given in Figure 8 (where the oriented edges are assigned counterclockwise order, and the edges of type-1 are marked by a cross $\times$). Figure 8(1) corresponds to a one-face imbedding of the bouquet $B_1$ (one vertex with one self-loop) on the projective plane. In particular, the face corners $c_1$ and $c_2$ in Figure 8(1) belong to the same face. Now suppose that we insert a new type-0 edge $e_2$ between these two face corners, as depicted in Figure 8(2). Our rules in §2 for pure graph rotation systems say that an edge insertion (necessarily type-0 for pure rotation systems) between two corners of the same face would split that face into two faces. However, when applying the face tracing procedure in Algorithm 1 to the graph of Figure 8(2), we find out that the resulting rotation system corresponds to a one-face imbedding of the bouquet $B_2$ (on the Klein bottle)!

![Figure 8: Inserting an edge into a general rotation system](image)

This phenomenon seems never explicitly described in the literature. A careful examination indicates that when tracing the face in Figure 8(1), the two face corners...
c₁ and c₂ are traversed in opposite directions (if corner c₁ is traversed in counter-clockwise order then corner c₂ would be traversed in clockwise order). Motivated by our observation of this phenomenon, we introduce the following new concept.

**Definition** Let ρ(G) be a general rotation system for a graph G. Each face corner (v, e, e') in ρ(G) is assigned a corner-type by the algorithm FbWalks(ρ(G)), as follows:

- (1). if FbWalks(ρ(G)) passes through vertex v by entering along edge e and leaving along edge e', then the corner (v, e, e') has corner-type 0;
- (2). if FbWalks(ρ(G)) passes through vertex v by entering along edge e' and leaving along edge e, then the corner (v, e, e') has corner-type 1;

We observe that if ρ(G) is a pure graph rotation system, then we can make all face corners type-0.

**Remark** The corner-types depend on how the algorithm call FbWalks(ρ(G)) picks an untraced corner (u, e, e') to start tracing the fb-walk for each new face F. There are only two ways to assign a first corner-type for a new face F: one is by FaceTrace(0, e') and the other by FaceTrace(1, e). By Corollary 3.2, the two ways to assign the first corner-type lead to exactly opposite corner types at every corner. The point is, for any two corners of F, either both ways assign the two corners the same type, or both ways assign the two corners different types. Therefore, since our concern is only whether they have the same or different types, it does not matter which way is used to assign the first corner-type for a face.

Our chief concern here is how surgery operations on a general graph rotation system impact the corresponding imbedding on a surface. At the outset of this section, we observed that this correspondence is non-obvious. In fact, for edge-insertions into general graph rotation systems, the rules given by the following theorem differ significantly from those for pure graph rotation systems. Since inserting a type-1 (i.e., twisted) edge can be implemented by inserting a type-0 (i.e., flat) edge followed by twisting the edge, we will consider here only the case of inserting type-0 edges.

### 4.1 Effects of edge-insertion surgery

**Theorem 4.1** Suppose that we insert the ends of a type-0 edge e into two face corners c₁ and c₂ in a general rotation system ρ(G). Then the following rules hold:

- (A) If the corners c₁ and c₂ belong to two different faces, then inserting edge e between c₁ and c₂ merges the two faces into a single face.
- (B) If the corners c₁ and c₂ belong to the same face, then
  - (B1) when c₁ and c₂ have the same corner-type, then inserting edge e between c₁ and c₂ splits the face into two faces;
  - (B2) when c₁ and c₂ have different corner-types, then inserting e between c₁ and c₂ results in a new face.

**Proof.** In case (A), since the corners c₁ and c₂ belong to different faces, the remark given just above this theorem allows us to assume that both corners have type-0. This reduces case (A) to inserting a type-0 edge between type-0 face corners of two different faces, which we know merges those two faces into a single face.

For case (B1), since the corners c₁ and c₂ have the same type, the same remark permits us to assume again that both corners have type-0. This reduces case (B1)
to the case of inserting a type-0 edge between two type-0 corners of the same face, which we know splits that face into two faces. A proof of cases (A) and (B1) for pure rotation systems appears in [FuGrMc88] (§2.4, page 526).

For case (B2), we may assume that the corner \( c_1 = (v_1, (v_1, w_1), (v_1, u_1)) \) of face \( F \) is type-0, and that the corner \( c_2 = (v_2, (v_2, u_2), (v_2, w_2)) \) of face \( F \) is type-1, by that same remark. Suppose that the new edge \([v_1, v_2]\) is inserted between corners \( c_1 \) and \( c_2 \). Under these assumptions, the fb-walk of the face \( F \) of the original rotation system \( \rho(G) \) is of the form

\[
F: \quad (\ast, (w_1, v_1)), (0, (v_1, u_1)), \alpha, (\ast, (w_2, v_2)), (1, (v_2, u_2)), \beta,
\]

where \( \ast \) represents trace-types that are irrelevant, and where \( \alpha \) and \( \beta \) are subsequences in the fb-walk. Note that \( (v_1, u_1) \) is 0-next to \( (v_1, w_1) \) at \( v_1 \) and that \( (v_2, u_2) \) is 1-next to \( (v_2, w_2) \) at \( v_2 \). See Figure 9(1) for an illustration. After inserting the new edge \([v_1, v_2]\), the resulting rotation system \( \rho'(G') \) (where \( G' \) is \( G \) plus the new edge) has the following new relations (see Figure 9(2)):

- (b1) \( (v_1, v_2) \) is 0-next to \( (v_1, w_1) \) at \( v_1 \);
- (b2) \( (v_1, u_1) \) is 0-next to \( (v_1, v_2) \) at \( v_1 \);
- (b3) \( (v_2, v_1) \) is 1-next to \( (v_2, w_2) \) at \( v_2 \);
- (b4) \( (v_2, u_2) \) is 1-next to \( (v_2, v_1) \) at \( v_2 \).

![Figure 9: Inserting an edge: case (B2)](image)

Now consider the tracing process \( \pi = \text{FaceTrace}(0,(v_1,u_1)) \) in the new rotation system \( \rho'(G') \). Since no other face corners are affected by the edge insertion, the process \( \pi \) follows the sequence of \( F \) (starting from \( (0,(v_1,u_1)) \)) until it traces the oriented edge \( (w_2, v_2) \):

\[
S_1: (0, (v_1, u_1)), \alpha, (\ast, (w_2, v_2)), 1,
\]

where the last “1” indicates the trace-type after tracing the oriented edge \( (w_2, v_2) \). Then because of (b3), the next oriented edge to be traced by \( \pi \) is \( (v_2, v_1) \), and after tracing \( (v_2, v_1) \), the trace-type is 1 (because the new edge \([v_1, v_2]\) has twist-type 0):

\[
S_2: (1, (v_2, v_1)), 1.
\]

By relation (b1), the next oriented edge to be traced by \( \pi \) is \( (v_1, w_1) \). By Corollary 3.2, in the original rotation system \( \rho(G) \), the process \( \text{FaceTrace}(1,(v_1,u_1)) \) will trace the reverse of \( \text{FaceTrace}(0,(v_1,u_1)) \) (which is \( F \)). Therefore, the process \( \pi \) will follow the reverse of \( F \) (starting from \( (1,(v_1,w_1)) \) and changing all corner types) until it traces the oriented edge \( (u_2, v_2) \):

\[
S_3: (1, (v_1, w_1)), \beta^r, (\ast, (u_2, v_2)), 0,
\]

where \( \beta^r \) is the reverse of \( \beta \). Note also that by Lemma 3.1, after tracing \( (u_2, v_2) \), the process \( \pi \) has a trace-type 0.
By relation (b4), the next oriented edge to be traced by the process \( \pi \) is \((v_2, v_1)\), and after tracing \((v_2, v_1)\), the process \( \pi \) has a trace-type 0 (because the edge \([v_1, v_2]\) has type-0):

\[
S_4: (0, (v_2, v_1)), 0.
\]

Now by relation (b2), the next edge must be \((v_1, u_1)\). It follows that the process \( \pi = \text{FaceTrace}(0, (v_1, u_3)) \) terminates and completes its tracing with an fb-walk that is the concatenation of the sequences \( S_1, S_2, S_3 \), and \( S_4 \) above, i.e., we obtain the sequence \( F' = (0, (v_1, u_1)), \alpha, (\ast, (w_2, v_2)), (1, (v_2, v_1)), (1, (v_1, w_1)), \beta', (\ast, (u_2, v_2)), (0, (v_2, v_1)). \)

Note that the face \( F' \) has used both trace-pairs for the new edge \([v_1, v_2]\) (i.e., \((0, (v_1, u_1))\) and \((1, (v_2, v_1))\) in the above sequence) plus the same trace-pairs used in \( F \). (By Corollary 3.3, we may say that a trace-pair \((t, \langle v, w \rangle)\) is used if either \((t, \langle v, w \rangle)\) is used or \((t', \langle v, w \rangle)\) is used, where \( t' = t + \text{tw-type}([v, w]) + 1 \mod(2) \)).

No other face corners are affected, and we conclude that the new set of fb-walks in case (B2) can be obtained from those for the original rotation system \( \rho(G) \) by replacing face \( F \) by the new face \( F' \). This completes the proof for case (B2).

The results of edge-insertion between face corners of different types in a general rotation system, i.e., case (B2) in Theorem 4.1, seem not to have been explicated heretofore in the literature.

### 4.2 Effects of edge-deletion surgery

Now we turn to edge-deletion on general rotation systems. Since deleting an edge \( e \) of twist-type 1 can be implemented by first twisting that edge and then deleting the resulting edge \( e \) of twist-type 0, it is sufficient to focus on deleting a type-0 edge.

**Theorem 4.2** Deleting a type-0 edge \( e \) from a general graph rotation system \( \rho(G) \) satisfies the following rules:

- **(A)** Suppose that the two trace-pairs induced by edge \( e \) belong to the fb-walks of two different faces of the imbedding. Then deleting edge \( e \) merges the two faces into a single face.

- **(B)** Alternatively, suppose that the two trace-pairs induced by \( e \) both belong to the fb-walk of the same face \( F \) in the imbedding.
  - **(B1)** If the two trace-pairs induced by edge \( e \) use different oriented edges, then deleting edge \( e \) splits the fb-walk of the face \( F \) into two closed walks, each of which is the fb-walk of a new face of the resulting imbedding.
  - **(B2)** If the two trace-pairs induced by edge \( e = [u, w] \) use the same oriented edge, then deleting edge \( e \) changes the fb-walk of face \( F \) into the fb-walk of a single new face.

**Proof.** As in Theorem 4.1, proofs for cases (A) and (B1) can be derived by modifying the proofs of the corresponding theorem for edge-deletion from a pure rotation system. See the rule **Edge-Delete-0** in §2. Thus, we need to prove the theorem only for case (B2). By the premises, we can assume that the fb-walk of face \( F \) is

\[
F : (0, \langle u, w \rangle), (0, \langle w, w_0 \rangle), \alpha, (\ast, \langle u_1, u \rangle), (1, \langle u, w \rangle), (1, \langle w, w_1 \rangle), \beta, (\ast, \langle u_0, u \rangle)
\]

as shown in Figure 10(1) (note that \([u, w] \) has twist-type 0), where \( \alpha \) and \( \beta \) are subsequences in the fb-walk and \( \ast \) stands for irrelevant trace-types.
Extended Graph Rotation Systems as a Model for Cyclic Weaving

Figure 10: Deleting an edge: case (B2)

In the above sequence, we have assumed the following:

- (b1) \(\langle u, w \rangle\) is 0-next to \(\langle u, u_0 \rangle\) at \(u\);
- (b2) \(\langle w, w_0 \rangle\) is 0-next to \(\langle w, w \rangle\) at \(w\);
- (b3) \(\langle u, w \rangle\) is 1-next to \(\langle u, u_1 \rangle\) at \(u\); and
- (b4) \(\langle w, w_1 \rangle\) is 1-next to \(\langle w, u \rangle\) at \(w\).

Therefore, after deleting the edge \([u, w] \) (see Figure 10(2)),

- (b5) \(\langle w, w_0 \rangle\) becomes 0-next to \(\langle w, w_1 \rangle\) at \(w\), by (b2) and (b4), and
- (b6) \(\langle u, u_0 \rangle\) becomes 1-next to \(\langle u, u_1 \rangle\) at \(u\), by (b1) and (b3).

Let \(\rho'(G')\) be the rotation system obtained by deleting edge \([u, w] \) from \(\rho(G)\), where \(G'\) is the graph \(G - [u, w]\). The tracing process \(\pi = \text{FaceTrace}(0, \langle w, w_0 \rangle)\) follows the fb-walk of face \(F\) (starting from \((0, \langle w, w_0 \rangle)\)) until it traces the trace-pair \((\ast, \langle u_1, u \rangle)\):

\[
S_1 : (0, \langle w, w_0 \rangle), \alpha, \ast, (\ast, \langle u_1, u \rangle), 1.
\]

The last 1 is the trace-type after tracing the subsequence \(S_1\). By (b6), the oriented edge \(\langle u, u_0 \rangle\) is 1-next to \(\langle u, u_1 \rangle\) at \(u\). Therefore, the next oriented edge to be traced is \(\langle u, u_0 \rangle\), and the process \(\pi\) follows the sequence produced by \(\text{FaceTrace}(1, \langle u, u_0 \rangle)\). By Lemma 3.1, this process will follow the reverse of \(F\) (starting from \((1, \langle u, u_0 \rangle)\)) until it traces the oriented edge \(\langle w_1, w \rangle\):

\[
S_2 : (1, \langle u, u_0 \rangle), \beta^r, \ast, \langle w_1, w \rangle), 0,
\]

where \(\beta^r\) is the reverse of the subsequence \(\beta\). After tracing the subsequence \(S_2\), the trace-type is 0. Now by (b5), the next edge to be traced is \(\langle w, w_0 \rangle\). Therefore, the process \(\pi = \text{FaceTrace}(0, \langle w, w_0 \rangle)\) terminates with the fb-walk that is the concatenation of \(S_1\) and \(S_2\):

\[
F' : (0, \langle w, w_0 \rangle), \alpha, \ast, \langle u_1, u \rangle), (1, \langle u, u_0 \rangle), \beta^r, \ast, \langle w_1, w \rangle).
\]

Since the face \(F'\) uses all trace-pairs of \(F\) except for the two induced by \([u, w]\), and since \([u, w]\) has been deleted from \(\rho(G)\), we conclude that the rotation system \(\rho'(G')\) can be obtained from the rotation system \(\rho(G)\) by replacing the face \(F\) by the face \(F'\). This completes the proof for case (B2).

As with Theorem 4.1, prior literature does not seem to include a proof of case (B2) in Theorem 4.2.

5 Extended Graph Rotation Systems

The extensions to the edge-twisting operation that are now to be described are crucial for our approach to modeling cyclic weaving on orientable surfaces. They are for specifying weaves, and not for graph imbeddings on surfaces.
Whereas labeling each edge with a number 0 or 1 is sufficient for specifying graph imbeddings, our edge-labels for modeling weaving are drawn from the entire set of integers. Whereas in classical topological graph theory, an edge is either untwisted or twisted, in the EGRS model, by way of contrast, its states with respect to twisting are in bijective correspondence with the integers. The EGRS model was introduced in [AkChGr09]. What appears in this section is essentially a reprise.

In a topological understanding of graph theory, tracing a twisted edge “reverses” the local orientation of the rotation system; accordingly, the result of double-twisting an edge is topologically equivalent to an untwisted edge. By way of contrast, in our model for cyclic weaving, the two trace-pairs induced by a type-0 edge are regarded as two parallel segments — and twisting the edge is interpreted as crossing the two segments. We are also interested in knowing which segment goes over and which one under at the crossing point, and by how many turns one segment is twisted around the other. In our model of cyclic weaving, double twisting an edge is not the same as leaving it untwisted. Figure 11 gives some intuitive illustrations for edge-twisting in terms of the above interpretation.

Figure 11: (1) an untwisted edge; (2) a clockwise twisted edge; (3) a counterclockwise twisted edge; (4) a double-clockwise twisted edge; (5) a double-counterclockwise twisted edge.

Comparing (2) with (3) and (4) with (5) in Figure 11 reveals that the direction in which we twist the edge is clearly relevant to which segment passes over the other. Motivated by this, we introduce the following definitions.

**Definition** An edge is $1^+$-twisted (resp. $1^-$-twisted) if it is obtained from a flat rectangular paper strip, in which the two longer sides of the rectangle are interpreted as the two trace-pairs induced by the edge, by fixing one end of the strip and twisting the other end in clockwise (resp. counterclockwise) direction by $180^\circ$.

See Figure 11(2)-(3) for an illustration. Note that this definition is independent of which end and which side of the paper strip is fixed. More generally, we say that an edge is $k^+$-twisted (resp. $k^-$-twisted) for an integer $k \geq 0$ if the edge can be obtained from an untwisted edge by $k$ consecutive $1^+$-twists (resp. $1^-$-twists).

**Definition** An extended rotation system for a graph $G$ is obtained from a pure rotation system by assigning a number $k$ of twists, with $k \in \mathbb{Z}$, to every edge of $G$.

We point out that in terms of graph imbeddings on surfaces, as studied in [GrTu87, GuSt85], a $k^+$-twisted or $k^-$-twisted edge is equivalent to an untwisted edge if $k$ is even, and equivalent to a normally twisted edge if $k$ is odd.
6 Cyclic Plain-Weaving on Surfaces

We now examine how graph rotation systems can be used to create woven images on orientable surfaces. In the rest of this paper, “surfaces” will always refer to orientable surfaces, unless explicitly specified otherwise. We begin with some precise definitions. Theorem 6.3, which provides the theoretical basis for cyclic plain-weaving, is the main new result of this paper.

**Definition** A cyclic weaving on a surface $S$ is an immersion $\sigma : \cup C \to S$ of the union of a finite set $C = \{c_1, \ldots, c_k\}$ of circles. We require that every point in the disjoint union has a neighborhood that is mapped homeomorphically. However, the images of two circles in $C$ may intersect on the surface $S$, and the image of a single circle may self-intersect. The number of intersection points of the circles on the surface is finite, and the number of pre-images of any point on $S$ under $\sigma$ is finite. Moreover, every intersection is a true intersection, rather than a tangency.

**Definition** The thickness of a cyclic weaving $\sigma$ is the maximum number of pre-images of a point $p$ in the surface $S$ under the mapping $\sigma$, taken over all $p \in \sigma(C)$. Moreover, we assume that for each point $p$ in the image $\sigma(C)$, the mapping $\sigma$ also specifies an ordering of the pre-images of $p$, so that for two circles who images intersect at $p$ we know which is over the other at the point $p$.

We will concentrate on a most common cyclic weaving structure, defined as follows:

**Definition** A cyclic weaving $\sigma : \cup C \to S$ is a cyclic plain-weaving if it satisfies the following conditions:
- Its thickness is at most 2;
- For each circle $c \in C$, suppose that $\langle p_1, p_2, \ldots, p_m \rangle$ is the ordered sequence of crossing points encountered when traversing $\sigma(c)$ on $S$. Then $\sigma(c)$ assigns these points alternatingly as over and under. That is, the closed curve $\sigma(c)$ weaves alternatingly over and under as it crosses other circles or itself.

We point out that there are also other popular weaving patterns, such as twill-weaving and satin-weaving [GrSh80]. In fact, our recent research (e.g., [ACCXG11]) shows that our EGRS model can also be used for these weaving structures.

### 6.1 The weaving genus of a link

A knot is defined as an imbedding of the circle $S^1$ in $\mathbb{R}^3$. A link is an imbedding of a disjoint union of one or more circles in 3-space $\mathbb{R}^3$, although usually the word knot is used when there is only one circle. In classical knot theory (e.g., see [Ad04, CrFo77, Mu08]), a knot or link is commonly represented by a projection onto the plane $\mathbb{R}^2$, as illustrated in Figure 12. The central concern of knot theory is deriving ways to decide from their representations whether two knots (or two links) are equivalent. That is, could one of them be deformed into the other?

![Figure 12: (a) A trefoil knot. (b) The Hopf link.](image)
For the case in which the surface $S$ is a sphere, a cyclic plain-weaving on $S$ is equivalent to what topologists call an alternating projection of a link. (See [Ad04].) To a topologist, the link itself is an imbedding of the set $C$ of circles in 3-space, or more formally, an equivalence class of imbeddings that are ambient isotopic to each other. As far as we know, the notion of alternating projections onto surfaces other than the sphere has not been well-developed by topologists. We briefly digress from weaving into some link theory.

**Proposition 6.1** For any link $L$ in 3-space $\mathbb{R}^3$ with $n$ components $c_1, \ldots, c_n$, there is a closed orientable surface of genus $n$ in $\mathbb{R}^3$ on which $L$ is imbedded.

**Proof.** Thicken each component $c_j$ into a solid torus, so that $c_j$ lies on the surface of that solid torus, and so that the solid tori are mutually disjoint. Next discard the interiors of the solid tori, so that each component of the link lies on a torus. Then connect the $n$ tori with $n-1$ tubes, to obtain a copy $S$ of the surface $S_n$ of genus $n$.

**Remark** A closed surface in $\mathbb{R}^3$ separates $\mathbb{R}^3$ into two parts, by a 3-dimensional analogue of the Jordan curve theorem. The part that extends to infinity is called the outside, and the other part is called the inside.

**Definition** Restoration of a link $L$ from a projection onto a surface is the result of pulling each crossing apart: a small over-crossing segment is pulled outside the surface and a small undercrossing segment is pushed inside the surface.

**Definition** An alternating projection of a link $L$ in $\mathbb{R}^3$ onto a surface $S$ is a continuous function $\mathbb{R}^3 \to S$ whose restriction to $L$ is a cyclic plain-weaving, whose restoration is equivalent to $L$ in $\mathbb{R}^3$.

**Corollary 6.2** Every link $L$ in 3-space has an alternating projection.

**Proof.** By Proposition 6.1, there is a closed orientable surface $S$ in $\mathbb{R}^3$ such that link $L$ is imbedded on $S$. An imbedding is an alternating projection with zero crossings. 

**Definition** The weaving genus of a link $L$ in $\mathbb{R}^3$ is the minimum genus taken over all closed surfaces in $\mathbb{R}^3$ onto which the link $L$ has a cyclic plain-weaving.

**Remark** For any integers $n$ and $g$ such that $n \geq 1$ and $0 \leq g \leq n$, it is possible to construct an $n$-component link with weaving genus $g$. Use $g$ copies of a non-trivial knot $K$ and $n-g$ copies of the trivial knot, such that each component is splittable from the others.

### 6.2 Constructing plain-weavings

Let $\pi_0 : G \to S$ be an imbedding of a graph on an oriented surface, and let $\rho_0(G)$ be the corresponding pure rotation system, whose rotations are taken to be counterclockwise around the vertices. We now present a method to construct a cyclic plain-weaving on $S$. From a topological perspective, this method could be described in terms of surgery on a regular neighborhood of the image of the graph $G$ in the surface $S$, in the form of a band-decomposition [GrTu87] of $S$. A combinatorial specification is essential for a computer-graphics implementation.

Each edge $[u, w]$ of $G$ corresponds to two trace-pairs $(0, \langle u, w \rangle)$ and $(1, \langle u, w \rangle)$ in $\rho_0(G)$ (as well as in $\pi_0(G)$). Note that by Corollary 3.3, these two trace-pairs
have two equivalent trace-pairs in terms of the oriented edge \( \langle w, u \rangle \). In general, for each trace-pair \((t_1, \langle u, w \rangle)\), we create a corresponding “segment” \( t_2 \langle u, w \rangle t_2 \), where \( t_2 = t_1 + \text{tw-type}(\langle u, w \rangle) \mod 2 \) is the trace-type we obtain if we start with the trace-type \( t_1 \) and trace the oriented edge \( \langle u, w \rangle \). The endpoints \( u \) and \( w \) are called the head and the tail of the segment, respectively.

To obtain a cyclic weaving \( \sigma_0 \) on the surface \( S \), we proceed as follows. For each edge \( [u, w] \) of \( \rho_0(G) \), we create the two segments \( 0 \langle u, w \rangle 0 \) and \( 1 \langle u, w \rangle 1 \) on \( S \) (note that \( \rho_0(G) \) is a pure rotation system). For the fb-walk of a face

\[
F : (0, \langle u_1, u_2 \rangle), \ldots, (0, \langle u_{h-1}, u_h \rangle), (0, \langle u_h, u_1 \rangle),
\]

we let the tail of the segment \( 0 \langle u_{i-1}, u_i \rangle 0 \) be connected to the head of the segment \( 0 \langle u_i, u_{i+1} \rangle 0 \) for all \( i = 2, \ldots, h + 1 \) (where we let \( h + 1 = 1 \) and \( h + 2 = 2 \)).

**Remark** We visualize the segment \( 0 \langle u, w \rangle 0 \) as lying slightly to one side of the edge \( [u, w] \) and the segment \( 1 \langle u, w \rangle 1 \) as lying slightly to the other side, as illustrated in Figure 13.

![Weaving structure changes corresponding to 1+-twist and 2+-twist](image)

Figure 13: Weaving structure changes corresponding to \( 1^+ \)-twist and \( 2^+ \)-twist

This is consistent with what we would obtain if we used the reverse of the fb-walk of \( F \) and the segments of the form \( 1 \langle u_i, u_{i-1} \rangle 1 \). Thus, the face \( F \) contains a circle \( c_F \) composed of the segments:

\[
c_F : 0 \langle u_1, u_2 \rangle 0, \ldots, 0 \langle u_{h-1}, u_h \rangle 0, 0 \langle u_h, u_1 \rangle 0.
\]  

\[ (6.2) \]
The resulting cyclic weaving algorithm 2 Segment-Twist.

**Algorithm 2** Segment-Twist.

**Segment-Twist**([u, w])

Input: an edge [u, w] with two parallel segments 0⟨u, w⟩0 and 1⟨u, w⟩1 in σ0.

Output: the edge [u, w] is k-twisted.

1. Cut the joint of 0⟨u, w⟩0 and 0⟨u, x⟩∗ and the joint of 1⟨u, w⟩1 and 1⟨w, y⟩∗ (as in Figure 13(A2) and Figure 13(B2));
2. Cross the segments 1⟨u, w⟩1 and 0⟨u, w⟩0 k times such that at the first cross point, the segment 1⟨u, w⟩1 goes above the segment 0⟨u, w⟩0 (as in Figure 13(A3) and Figure 13(B3));
3. If k is odd, then rename 0⟨u, w⟩0 as 0⟨u, y⟩1 and rename 1⟨u, w⟩1 as 1⟨w, y⟩0 (as in Figure 13(A4))
4. Connect the tail of *⟨u, w⟩1 with the head of 1⟨w, y⟩∗, and connect the tail of 0⟨u, w⟩0 with the head of 0⟨w, x⟩∗ (as in Figure 13(A5) and Figure 13(B4)).

The resulting cyclic weaving σ0' on the surface S is said to be induced by the extended rotation system ρ0(G).

The weaving structure change corresponding to a k−-twist of an edge in the pure rotation system ρ0(G) can be described similarly, except for Step 2, where we require after crossing the segments, that the segment 0⟨u, w⟩0 go above the segment 1⟨u, w⟩1 at their first crossing point. We may apply the edge-twisting operation on more than one edge in the pure rotation system ρ0(G). Figure 14 provides an example to illustrate the above idea.

Figure 14(a) is an orientable surface S (i.e., the sphere). Figure 14(b) shows an imbedding π0(G) of a graph G on the surface S (the graph edges are given as flat bands). Figure 14(c) is the pure rotation system ρ0(G) of G corresponding to the imbedding π0(G) in Figure 14(b). Figure 14(e) shows the initial cyclic weaving σ0 induced from the rotation system ρ0(G) in Figure 14(c). Figure 14(g) applies the
1\+\-twisting operation on three of the edges in the rotation system \(\rho_0(G)\), resulting in an extended rotation system \(\rho(G)\). Figure 14(j) is the cyclic weaving \(\sigma\) induced from the extended rotation system \(\rho(G)\).

We say that an edge is *twisted positively* if it is \(k^+\)-twisted for an integer \(k \geq 1\), and that it is *twisted negatively* if it is \(k^-\)-twisted for an integer \(k \geq 1\). The theorem below is a foundation for cyclic plain-weaving.

**Theorem 6.3** Let \(\rho_0(G)\) be a pure rotation system for an imbedding \(\pi_0 : G \rightarrow S\) of a graph on an orientable surface. Let \(A\) be an arbitrary subset of edges of \(G\). If we either twist all edges in \(A\) positively or twist all edges in \(A\) negatively, then the resulting extended rotation system induces a cyclic plain-weaving on \(S\).

**Proof.** We adopt the same notations as before. Let the cyclic weaving induced by the pure rotation system \(\rho_0(G)\) be \(\sigma_0\), where for each face \(F\) in \(\pi_0(G)\) there is a corresponding circle \(c_F\) in \(\sigma_0\) mapped to the surface \(S\), such that no two such circles intersect on \(S\). We provide a detailed proof of the theorem for the case where all edges in the subset \(A\) are twisted positively, by induction on the number \(h\) of edges in \(A\). The case where all edges in \(A\) are twisted negatively can be proved similarly.

For the subset \(A\) of \(h\) edges, let \(\rho_h(G)\) denote the extended rotation system obtained by positively twisting all \(h\) edges in \(A\) in the given pure rotation system \(\rho_0(G)\). We may regard \(\rho_0(G)\) as an extended rotation system in which every edge is 0-twisted.

For \(h \geq 1\), we choose an edge \([u, w]\) in \(A\) that is \(k^+\)-twisted in \(\rho_h(G)\) for some integer \(k > 0\), and we let \(\rho_{h-1}(G)\) be the extended rotation system obtained by replacing the \(k^+\)-twisted edge \([u, w]\) in \(\rho_h(G)\) by an untwisted edge \([u, w]\). Thus, \(\rho_{h-1}(G)\) can be obtained from \(\rho_0(G)\) by positively twisting the \(h - 1\) edges in \(A - \{[u, w]\}\), and \(\rho_h(G)\) can be obtained from the rotation system \(\rho_{h-1}(G)\) by \(k^+\)-twisting the edge \([u, w]\).

Let \(\sigma_h\) and \(\sigma_{h-1}\) be the cyclic weavings induced by the extended rotation systems \(\rho_h(G)\) and \(\rho_{h-1}(G)\), respectively.

**Claim 1.** Every segment in the cyclic weaving \(\sigma_h\) is of the form 

t_1([v_1, v_2]_{t_2}), \quad \text{where} \quad [v_1, v_2] \text{ is an edge in the graph } G, \quad \text{where} \quad t_1, t_2 \in \{0, 1\} \text{ are trace-types such that} \quad t_2 = t_1 + \text{tw-type}([v_1, v_2]) \pmod 2, \text{and where} \quad \text{tw-type}([v_1, v_2]) = 1 \text{ if } [v_1, v_2] \text{ is } j^+\text{-twisted in } \rho_h(G) \text{ for an odd number } j \text{ or is } 0 \text{ if } [v_1, v_2] \text{ is untwisted or } j^+\text{-twisted for an even number } j. 

Claim 1 is obviously true for \(h = 0\): all edges in \(\rho_0(G)\) are untwisted, and all segments in \(\sigma_0\) are of the form \(0([v_1, v_2])_0\) and \(1([v_1, v_2])_1\). By way of induction, we suppose that for some \(h > 0\), Claim 1 is true for the weaving \(\sigma_{h-1}\). The rotation system \(\rho_h(G)\) is obtained from \(\rho_{h-1}(G)\) by \(k^+\)-twisting the edge \([u, w]\). By the
procedure Segment-Twist, only the two segments \( o(u, w)_0 \) and \( 1(u, w)_1 \) in \( \sigma_{h-1} \) may change (note that the edge \([u, w]\) in \( \rho_{h-1}(G) \) is untwisted). By step 3 of the procedure Segment-Twist, these two edges are unchanged if \( k \) is even, and they change to \( o(u, w)_1 \) and \( 1(u, w)_0 \) if \( k \) is odd. Therefore, Claim 1 still holds true for the resulting weaving \( \sigma_h \) induced by the extended rotation system \( \rho_h(G) \). This proves Claim 1.

**Claim 2.** Let \([u, w]\), \([w, x]\), and \([w, y]\) be edges in \( G \) such that in the rotation at \( w \) in \( \rho_0(G) \), the oriented edge \langle w, x \rangle \) is 0-next to \langle w, u \rangle, and the oriented edge \langle w, y \rangle \) is 1-next to \langle w, u \rangle. Then

1. the tail of the segment \( u, w \) in \( \sigma_h \) is connected with the head of \( o(w, x) \); and
2. the tail of the segment \( u, w \) in \( \sigma_h \) is connected with the head of \( 1(w, y) \).

Claim 2 holds true for \( h = 0 \), by the specifications (6.1) and (6.2) for constructing the fb-walks. We assume inductively that for some \( h > 0 \), Claim 2 holds true for the weaving \( \sigma_{h-1} \). The weaving \( \sigma_h \) is obtained from the weaving \( \sigma_{h-1} \) by modifying only the segments \( o(u, w)_0 \) and \( 1(u, w)_1 \), corresponding to the \( k^* \)-twist of the edge \([u, w]\) according to the procedure Segment-Twist. By the induction hypothesis on \( \sigma_{h-1} \) and by Step 4 of the procedure Segment-Twist, Claim 2 still holds true for the weaving \( \sigma_h \). This proves Claim 2.

**Claim 3.** There is a one-to-one correspondence between the fb-walks in the extended rotation system \( \rho_h(G) \) and the circles in the weaving \( \sigma_h \).

Let \( t_1(v_1, v_2)_{t_2} \) and \( t_3(v_3, v_4)_{t_4} \) be two consecutive segments of a circle \( c \) in the weaving \( \sigma_h \). Claim 1 implies that \( t_2 = t_1 + tw\text{-type}([v_1, v_2]) \mod 2 \) and that \( t_4 = t_3 + tw\text{-type}([v_3, v_4]) \mod 2 \). Claim 2 implies that \( t_2 = t_3 \) and \( v_2 = v_3 \), and that \( [v_3, v_4] \) must be the \( t_2 \)-next to \( [v_2, v_1] \) in the rotation at \( v_3 = v_2 \). Therefore, if we start with the trace-pair \( (t_1, [v_1, v_2]) \) in the rotation system \( \rho_h(G) \) and apply the subroutine FaceTrace in Algorithm 1, the next trace-pair must be \( (t_3, [v_3, v_4]) \). Therefore, the circle \( c \) in \( \sigma_h \) must follow exactly tracing an fb-walk in the rotation system \( \rho_h(G) \). Conversely, using the same reasoning, we can prove that an fb-walk

\[
F : (t_1, [u_1, u_2]), (t_2, [u_2, u_3]), \ldots, (t_p, [u_p, u_1])
\]

in the rotation system \( \rho_h(G) \) induces the circle

\[
c_F : t_1(v_1, u_2)_{t_2}, t_3(v_2, u_3)_{t_3}, \ldots, t_p(u_p, u_1)_{t_1},
\]

in the weaving \( \sigma_h \). This proves Claim 3.

**Claim 4.** Let \([v_1, v_2]\) be any edge of the graph \( G \). Then

1. if we traverse a circle \( c \) in \( \sigma_h \) starting from the head of the segment \( o(v_1, v_2)_0 \), in \( c \), then the circle \( c \) must go “under” at the first crossing point;
2. if we traverse a circle \( c \) in \( \sigma_h \) starting from the head of the segment \( 1(v_1, v_2)_0 \), in \( c \), then the circle \( c \) must go “over” at the first crossing point;
3. if we traverse a circle \( c \) in \( \sigma_h \), and if the last segment in \( c \) is \( *v_1, v_2)_0 \), then at the last crossing point before we reach the tail of \( *v_1, v_2)_0 \), the circle \( c \) must go “over”;
4. if we traverse a circle \( c \) in \( \sigma_h \), and if the last segment in \( c \) is \( *v_1, v_2)_1 \), then at the last crossing point before we reach the tail of \( *v_1, v_2)_1 \), the circle \( c \) must go “under”.

Claim 4 is trivially true for the case \( h = 0 \) because the initial weaving \( \sigma_0 \) corresponding to the pure rotation system \( \rho_0(G) \) has no circle crossings. For \( h > 0 \), the weaving \( \sigma_h \) is obtained from \( \sigma_{h-1} \) by modifying the segments \( o(w, w)_0 \) and \( 1(w, w)_1 \) in \( \sigma_{h-1} \), corresponding to the \( k^+ \)-twist of the edge \([u, w]\) and using the procedure Segment-Twist. Now suppose inductively that Claim 4 holds true on the weaving \( \sigma_h \) for all edges in \( G \), except perhaps for the edge \([u, w]\). For the edge \([u, w]\), we note that the two new segments \( o(w, w)_* \) and \( 1(w, w)_* \) in \( \sigma_h \) cross each “over” and “under” alternatingly, such that (see Figure 13 for verifications):

- a1. if we traverse in \( \sigma_h \) the new segment \( o(w, w)_* \) from its head, then at the first crossing point with the segment \( 1(w, w)_* \), we go “under”;
- a2. if we traverse in \( \sigma_h \) the new segment \( 1(w, w)_* \) from its head, then at the first crossing point with the segment \( o(w, w)_* \), we go “over”;
- a3. if we traverse in \( \sigma_h \) the new segment \( 1(w, w)_1 \) and reach its tail, then at the last crossing point with the segment \( o(w, w)_0 \), we went “over”;
- a4. if we traverse in \( \sigma_h \) the new segment \( 1(w, w)_0 \) and reach its tail, then at the last crossing point with the segment \( o(w, w)_1 \), we went “under”.

Therefore, Claim 4 also holds true for the edge \([u, w]\) after it is \( k^+ \)-twisted in the weaving \( \sigma_h \). This proves Claim 4.

Now we are ready to prove the theorem. Pick any circle \( c \) in the weaving \( \sigma_h \). By Claim 3, there is an \( \text{fb-walk} \)

\[
F_c : (t_1, (u_1, u_2)), (t_2, (u_2, u_3)), \ldots, (t_p, (u_p, u_1))
\]  

(6.3)

in the rotation system \( \rho_h(G) \) such that the circle \( c \) is given as a sequence of segments of the form:

\[
c : t_1 (u_1, u_2) t_2, t_2 (u_2, u_3) t_3, \ldots, t_p (u_p, u_1) t_1.
\]  

(6.4)

Pick any two consecutive segments \( t_{i-1} (u_{i-1}, u_i)t_i, t_i (u_i, u_{i+1})t_{i+1} \) in the circle \( c \) as given by (6.4) (we have let \( p+1 = 1 \) and \( p+2 = 2 \)). There are two possible subcases:

Subcase 1: \( t_i = 0 \). By Claim 4(3), at the last crossing point before we reach the tail of \( t_{i-1} (u_{i-1}, u_i)t_0 \), the circle \( c \) went “over”, and by Claim 4(1), at the first crossing point after the head of \( o(u_i, u_{i+1})t_{i+1} \), the circle \( c \) will go “under”.

Subcase 2: \( t_i = 1 \). By Claim 4(4), at the last crossing point before we reach the tail of \( t_{i-1} (u_{i-1}, u_i)t_1 \), the circle \( c \) went “under”, and by Claim 4(2), at the first crossing point after the head of \( 1(u_i, u_{i+1})t_{i+1} \), the circle \( c \) will go “over”.

Therefore, no matter which is the case, the sub-chain \( t_{i-1} (u_{i-1}, u_i)t_i, t_i (u_i, u_{i+1})t_{i+1} \) traverses the crossing points “over” and “under” alternatingly. Since each segment \( t_{i-1} (u_{i-1}, u_i)t_i \) in the circle \( c \) also traverses the crossing points with its “partner” segment \( t'_{i-1} (u_{i-1}, u_i)t'_{i} \) “over” and “under” alternatingly, where \( t'_{i-1} = t_{i-1} + 1 \ (\text{mod} \ 2) \) and \( t'_{i} = t_i + 1 \ (\text{mod} \ 2) \), we conclude that the circle \( c \) in \( \sigma_h \) traverses its crossing points “over” and “under” alternatingly.

Since \( c \) is an arbitrary circle in the weaving \( \sigma_h \), all that remains to prove is that the thickness of \( \sigma_h \) is at most 2 — but that is clear: each crossing point in \( \sigma_h \) is made by a pair of segments \( o(w, w)_* \) and \( 1(w, w)_* \) for an edge \([u, w]\) in \( G \), and such a crossing has thickness at most 2. This completes the proof for the theorem. \( \square \)
7 Conclusion

In this paper, we have provided precise descriptions and characterizations for the fundamental surgery operations on general graph rotation systems, including edge-twisting, edge-insertion, and edge-deletion. We have shown that these surgery operations behave differently on general graph rotation systems from how the corresponding operations act on pure graph rotation systems. Theorem 3.4, Theorem 4.1, and Theorem 4.2 refine the existing understanding of how the fb-walks are changed when these edge-surgery operations are applied to the corresponding rotation system.

Based on this newly precise theoretical underpinning, we have proposed a new paradigm for constructing cyclic weaving structures. In particular, we prove that our extended graph rotation systems provide a very simple and effective way for specifying cyclic plain-weaving structures on orientable surfaces. Moreover, the edge-surgery operations on the corresponding graph rotation systems provide a set of powerful tools for effectively handling changes in the weaving structures dynamically. Figure 1 shows some examples of the cyclic weaving structures created by our software, which is based on the theoretical study presented in the current paper.

Our theory and techniques are applicable to any graphs. In particular, they can be applied to all those that are used in mesh modeling in computer graphics and computer-aided design. Therefore, given a mesh structure on a surface (i.e., an orientable 2-manifold), one will be able to apply our techniques on the mesh and construct a woven pattern on the surface. It should be noted that the methods provide very efficient constructions of woven patterns, because they are based on graph rotation systems, which have been successfully used in efficient surface modeling [AkCh99]. Moreover, our results can be used to characterize the structures of the woven pattern. In [AkChGr09], we have described all necessary details to show how to apply our theory and techniques on triangulated meshes on surfaces, which are the most popular mesh structure used in surface modeling.

The approach we have described here has been implemented in a 3D-modeling software system called TopMod3D, which serves as proof-of-concept. After it originated as a by-product of some thesis and class projects such as [AkChSr00, ACSE01], TopMod was refined for use by 3D-graphic artists. The initial version, called TopMod 1.0, has been available as free software since 2003. The second version, TopMod 2.0 — with an improved user interface and scripting editor, was released in August 2007. Since 2003, more than 50,000 people have downloaded the software, and many talented artists have used it to create interesting sculptures [ACSMT08].

References


Extended Graph Rotation Systems as a Model for Cyclic Weaving


Authors’ addresses:

• E. Akleman, Department of Visualization, Texas A&M University, College Station, TX 77843, email: ergun@viz.tamu.edu;

• J. Chen, Department of Computer Science, Texas A&M University, College Station, TX 77843, email: chen@cs.tamu.edu;

• J. L. Gross, Department of Computer Science, Columbia University, New York, NY 10027, email: gross@cs.columbia.edu.