ACM Symposium on Solid Modeling and Applications (2004)
G. Elber, N. Patrikalakis, P. Brunet (Editors)

Shortest circuits with given homotopy in a constellation

Dominique Michelucci and Marc Neveu

University of Burgundy, Dijon, France

Abstract

A polynomial method is described for computing the shortest circuit with a prescribed homotopy on a surface. The surface is not described by a mesh but by a constellation: a set of sampling points. Points close enough (their distance is less than a prescribed threshold) are linked with an edge: the induced graph is not a triangulation but still permits to compute homologic and homotopic properties. Advantages of constellations over meshes are their simplicity and robustness.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Computational Geometry and Object Modeling

1. Introduction

This self contained article describes a method to compute the shortest circuit with prescribed homotopy, lying on a given surface. This problem was met with some industrial contract, to simulate and optimize the shape of electric wiring. A mould surface is given, with obstacles (convex polyhedra, for short). Ignoring obstacles, the surface has initially the topology of a disk, or a cylinder, or a sphere, and any geometry (shape) compatible with this topology. The geometry is described by a given triangular mesh. A slack circuit on the surface is also given; it typically turns around some obstacles to avoid them. The shortest circuit with the same homotopy (turning around the same obstacles and in the same order) is computed with the method presented here (Fig. 1 and 3). Then electric wires are taut along this shortest circuit: their shape is computed and approximated by convex polyhedra; they create new obstacles, which modify the topology of the mould surface. About sixty shortest circuits are computed; our program needs 5 to 10 minutes on a standard PC.

We first represented the free region on the mould surface with a triangular mesh but we faced terrible robustness issues [BMP94], for instance when updating the free region with new obstacles. Actually, the initial mesh was not always consistent. We then realized that the mesh is useless, and that a constellation is sufficient, simpler and robust: all robustness problems disappear.

To get a constellation from the triangular mesh of the mould surface, each triangle is sampled with points. Some uniform random distribution is assumed (from 0.1 to 1.0 point per square millimeter). Each time two sampling points are distant by less than a prescribed threshold R, and the segment they define does not cut any obstacle, they are linked by an edge, the cost of which is the distance between the two points. This induces a non oriented graph, where edges carry positive weights. This graph still permits to compute shortest tours, and updating the BRep of the free area on the mould surface becomes useless: instead, it is sufficient to

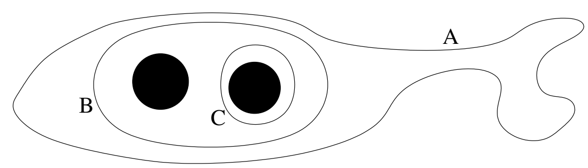

Figure 1: *Circuits A and B are equivalent. A and C are not.*

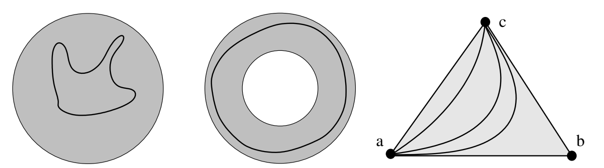

Figure 2: *In a disk, the shortest tour equivalent to the slack one has length 0. In the annulus, it is the inner cercle. Right: how the wire ac can continuously glide in the triangle abc to eventually coincide with ab, bc.*

test whether each edge between two close sampling points cuts a new obstacle or not. The graph is quite large, and some bucketing technique is used for optimization, as usual.

With sensors, cloud of points emerge as a basic and ubiquitous data structure in CADCAM; rendering in Computer Graphics and NC machining also use it; thus it is interesting to see how far we can go with such a simple data structure.

E. Colin de Verdière and F. Lazarus investigate the same questions [dVL03], but consider triangular meshes. Numerous papers deal with shortest paths on surfaces or meshes [Mit97], but computations of shortest tours with given homotopy is less investigated up to now. As far as we know, this is the first article which deals with this problem with constellations. Homology and homo-

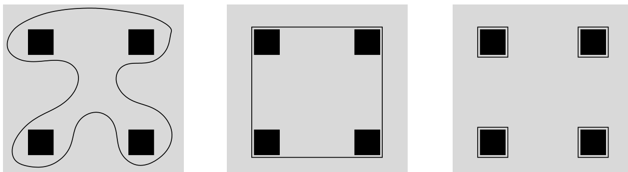

Figure 3: *Left: a slack tour around 4 black square obstacles. Middle: the shortest homotopic tour. Right: the shortest homologic tour is composed of 4 squares around the 4 obstacles.*

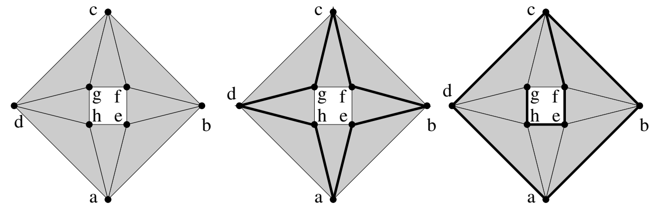

Figure 4: *Bold edges, middle and right, give possible bases. abcda and $efghe$ are homologic circuits, non zero (they cannot be contracted into a single point). They are also homotopic.*

topy are standard concepts in algebraic or combinatorial topology [FK98, Veg97] but the latter always considers cellular complexes.

Section 2 gives basic definitions. Section 3 and 4 explain how to compute a base of the constellation to test homology and homotopy between circuits. Section 5 defines the disk and cuts decomposition, the unfolded graph, round trips and shortcuts, and presents our method. Section 6 illustrates by an example the difference between homology and homotopy. Section 7 concludes.

2. Definitions

Fig. 1 shows equivalent and non equivalent tours. In a disk, all shortest tours have null length because no obstacle inside the tour prevents to contract it into a single point (Fig. 2 left). In an annulus, some tours are not contractible into a single point, because of the hole (an obstacle), see Fig. 2 middle; thus the smallest equivalent tour is the inner circle. Fig. 3 shows there are two definitions for the shortest tour: the homotopic one and the homologic one. In our application, we need the homotopic one. They are now defined.

Three vertices a, b, c of a constellation are a triangle iff the three edges exist in the graph. It is assumed (simplicity axiom) that *every triangle abc in the graph also exists in the surface* in the following sense: a wire (a,c) from a to c lying on the underlying surface can be smoothly deformed into (a,b) followed by (b,c), without leaving the surface (see fig. 2, right). This property can be noted in two ways, first an homologic notation, second an homotopic notation.

In homology, each triangle a, b, c defines a linear relation: $x_{a,c} = x_{a,b} + x_{b,c}$, where each arc i, j in the graph has a corresponding symbol $x_{i,j}$; moreover, for each edge i, j, the relation $x_{i,j} = -x_{j,i}$ holds. Triangles and edges give a system of linear equations on arc symbols. The empty set is noted 0; for instance $x_{i,j} + x_{j,i} = 0$. With Gauss elimination, or some other method such as LU decomposition, it is possible to compute a base of arc symbols; all arc symbols can then be expressed as a linear combination of the basic arc symbols; coefficients in linear combinations are rational numbers; actually they are integers 0, ± 1, apart in exotic cases (similar to non eulerian polyhedra or non manifold objects) which are not discussed here.

Homotopy preserves order, contrarily to homology. Each triangle a, b, c defines a word relation: $x_{a,c} = x_{a,b}x_{b,c}$. Homotopy uses the (non commutative) product, or concatenation, as notation, instead of the (commutative) additive notation used by homology. Each edge u, v gives two inverse (instead of opposite) symbols $x_{u,v}$ and $x_{v,u}$. We use the notations: $x_{u,v} = x_{v,u}^{-1} = \overline{x_{v,u}}$ and $x_{u,v}x_{v,u} = \varepsilon$. ε is the neutral element for concatenation. Two sequences α and β are inverse when $\alpha\beta = \varepsilon$; for instance each triangle a, b, c gives a circuit homotopic to ε: $x_{a,b}x_{b,c}x_{c,a} = \varepsilon$. The only sequences used in this paper are paths or circuits: for two contiguous symbols, the right index (a vertex in the graph) of the first symbol and the left one of the second symbol (another vertex) are equal.

Other examples: two triangles a,b,c and a,b,c' contiguous along a,b give a circuit $x_{a,b}x_{b,c}x_{c,b'}x_{b',a} = (x_{a,b}x_{b,c})(x_{c,b'}x_{b',a}) = (x_{a,c})(x_{c,b'}x_{b',a}) = (x_{a,c})(x_{c,a}) = \varepsilon$ homotopic to ε, by using the two triangular relations, and an edge relation. On the other hand, not all circuit are homotopic to ε; for instance the square a,b,c,d (the graph contains edges (a,b), (b,c), (c,d), (d,a), but not the edge (a,c) nor (b,d)) is not homotopic to ε, because no triangular relation permits to reduce the sequence $x_{a,b}x_{b,c}x_{c,d}x_{d,a}$. The simplicity axiom implies that all circuits non homotopic to ε contain strictly more than three edges of the graph.

Figure 4 shows a simple constellation (it is even a triangulation), equivalent to an annulus. Circuits *abcda* and $efghe$ are homologic, because they express the same in some base, for instance the base in the middle part of figure 4. Both circuits *abcda* and $efghe$ express as: $(-be+bf)+(-cf+cg)+(-dg+dh)+(-ah+ae)$.

Homology is weaker than homotopy. For instance sequences $\alpha\beta\overline{\alpha}\overline{\beta}$ is not homotopic to ε, whereas it is homologic to 0 : $\alpha+\beta-\alpha-\beta = 0$. Two homotopic sequences are always homologic; the converse is wrong. All bases for homology give a base for homotopy. A base can be computed with Gauss elimination method in the (homology) linear system, in $O(m^3)$ time, where m is the number of arcs. Another method is proposed in section 3.

3. Computation of a base

To test homotopy between two given circuits, the naive method, which explores all possible transitions, has an exponential cost. A polynomial solution is to first compute a base. All edges can then be expressed with basic edges only (for homology and homotopy). Two circuits are homologic iff their decompositions (two linear combinations in a vectorial space) are equal. Two circuits are homotopic iff their sequences are equal up to some circular permutation.

To compute a base, compute first a covering tree of the graph. The graph is assumed to be connected, without loss of generality. Any covering tree can be used. By definition it contains no triangle, thus all its edges are homotopy (homology) independent.

Any set of edges generate, by homotopic (homologic) relations of edges and triangles, a superset of edges called its closure. To compute the closure of a given generating set, the closure is initialized with the generating set. Then, each time an edge ac is the

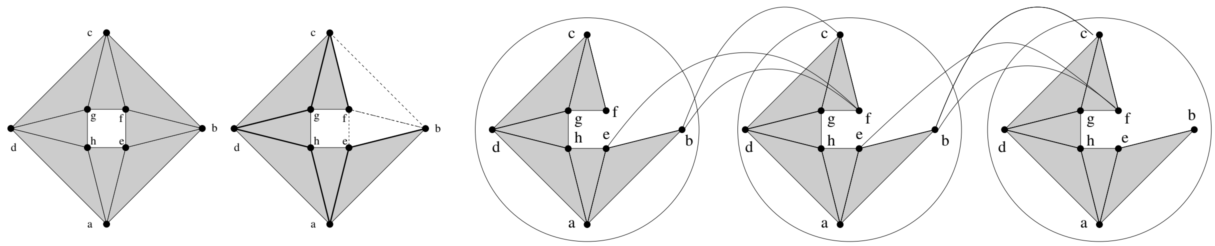

Figure 5: *The constellation; a covering tree and its disk; a part of the unfolded graph.*

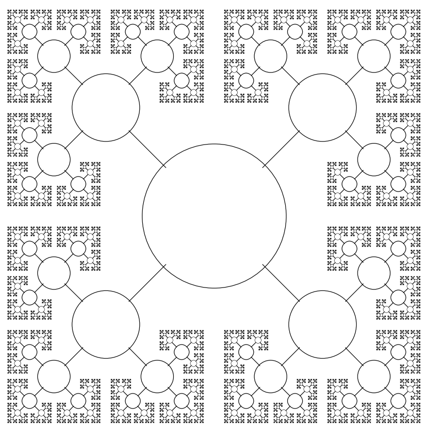

Figure 6: *An unfolded graph, with two cuts.*

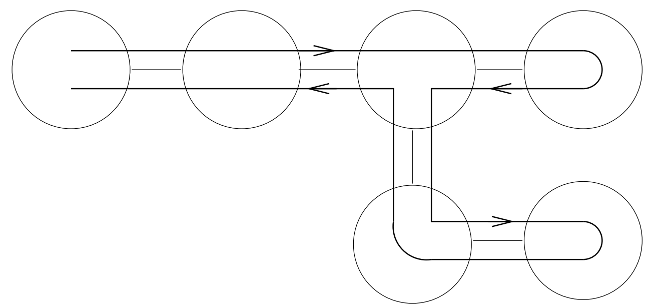

Figure 7: *A round trip is a path in the unfolded graph which comes back to the copy it started from.*

third edge of some triangle *abc* whose two other edges *ab* and *bc* are already in the closure, the edge *ac* is inserted in the closure. The closure of the covering tree is called the "disk" because all its circuits are homotopic to ε, as circuits in a topologic disk (Fig. 2 left). Later, the disk will be graphically displayed as a disk (fig. 6). If all edges are in the disk, *ie* the closure of the covering tree, the method is terminated: all circuits are homotopic to ε, and the optimal circuit with prescribed homotopy (it can only be ε) is the null circuit, with null cost. Otherwise, while there is an edge outside the closure of the current base, it is inserted in the current base, and the corresponding closure is updated. It means the latter edge is inserted in the closure, together with all third edges of triangles whose two other edges are already in the closure. In other words, each time an edge *uv* is inserted in the closure, we have to "propagate" it: we consider all triangles *uvw* (all triangles which contain the *uv* edge); if *uw* is in the closure and *vw* is not, then *vw* is inserted in the closure, and propagated. Propagation is managed as usual with some stack, and stops when the stack is empty. Each edge is propagated once (just after it is inserted in the closure). Finding all triangles with a given edge *uv* can be done in time proportional to the degrees of *u* and *v*. The sum of all degrees is $O(m)$ where m is the number of edges in the graph. Thus the construction of the base is in $O(m)$. In constellations (as in triangulations), m is $O(n)$, the number of vertices, but the average degree is greater for constellations (30 or 60).

The base is composed of a covering tree, plus h edges, one edge per independent tour (a *tour* is a circuit not homotopic to ε). The homologic relations are just a system of linear equations, making obvious the matroidal [CLR90, PS98] structure and properties of homology. As a consequence, for a given graph, all homologic bases have equal cardinality (the so called rank of the matroid). All covering trees have also equal cardinality (covering trees are bases of another matroid, called graphic matroid). Thus the difference between these two numbers (h) is independent of the used covering trees and homologic bases. Actually all (accurate enough) constellations for the same surface and obstacles yield the same value for h, while their number of edges and vertices can be very different.

4. Expressing edges in the base

It is also possible to compute the homologic expression of edges in the base, while constructing the base. For homology, the expression of a basic arc *uv* is $x_{u,v}$, and the opposite arc $x_{v,u}$ has definition $-x_{u,v}$. Each time an edge *uw* is inserted in the closure because it is the third edge of a triangle *uvw* whose two other edges already belong to the closure, the definition of *uw* is the sum of the definition of *uv* and the one of *vw*, which are already known. Clearly, computation of all expressions is $O(m(n-1+h))$ time, using a vectorial representation for definitions: n is the number of vertices, a covering tree has thus $n-1$ edges, thus the base has cardinality $n-1+h$ (h is the number of independent tours). In practice, the matrix of definitions is sparse, and this measure $O(m(n-1+h))$ is very pessimistic.

The same method can be used to compute homotopy definition sequences, for each edge (if arc *uv* has definition α, arc *vu* has inverse definition α^{-1}, also noted $\overline{\alpha}$). Each time an edge *uw* is inserted in the closure because it is the third edge of a triangle *uvw* whose two other edges already belong to the closure, the definition of *uw* is the concatenation (instead of the sum for homology) of the definition of *uv* and the one of *vw*, which are already known. Actually, some reduction step is also needed, to eliminate contiguous inverse terms; for instance $xyz\bar{z}\bar{y}$ must be reduced to x. Reduction uses a stack, initially empty. The sequence to be reduced is read from left to right, one symbol at a time; if the stack is empty, or if the top of the stack and the current symbol are not inverse, the current read symbol is pushed on the stack, otherwise the top symbol is removed from the stack and the current read symbol is discarded (since these two symbols annihilate each other). Eventually the stack contains the reduced sequence (in reverse order). Finally, homotopy sequences for circuit must also be "circularly reduced": while the first and the last symbol are inverse, they must be removed. Then, two circuits with (reduced and circularly reduced) homotopy sequences s and t are homotopic iff s and t have equal length and ss contains t (see [CLR90] for algorithms on words). In practice, the time for computing homotopy definitions is the same as the one for computing homology definitions.

5. Shortest homotopic tour

5.1. An example

Fig. 8 shows a simple example. For readability, the euclidean plane is sampled with a regular grid; there are four square (unsampled) holes or obstacles, thus $h = 4$. Top subfigure displays in thick lines an initial circuit around the upper left hole. Middle part displays one of the numerous possible disks. Bottom displays in thick lines the shortest homotopic circuit (it is also homologic). In real world applications, sampling is more random, the average degree of vertices is much greater (30 or 60, instead of 8 in this trivial example), and there are several thousand points.

5.2. Principles

Some circuit is given, and one wants the shortest circuit with the same homotopy. The method presented here computes a base, and partitions the edges of the graph into several sets: the disk and h cuts. The disk is the closure of the used covering tree; the h edges in the base which do not belong to the covering tree are basic cut edges.

To simplify, it is assumed that the definition sequence of each edge depend only of at most one basic cut edge, and of an arbitrary number of basic disk edges. Section 5.4 explains how insertion of auxilliary vertices reduces complex cases to this simple one.

An enlightening idea is the one of unfolding the graph. Figure 5 shows from left to right a constellation of an annulus ($h = 1$), a covering tree and its disk (cut edges are dotted), a part of the unfolded graph. The unique cut is composed by arcs bf, ec, ef, the inverse cut by arcs fb, ce, fe. To get the entire unfolded graph, the central copy of the figure must be repeated an infinity of times, at left and right. Note the asymmetry of cut edges between two contiguous copies: edges are bf, ec, ef (the first vertex lie in the copy on the left, the second in the copy on the right), and not fb, ce, fe.

Figure 6 shows the structure of an unfolded graph for a constellation with $h = 2$ basic cut edges α and β. The disk (represented as a disk) is copied an infinity of times. Two contiguous copies are linked either by a cut edge α (representing all edges depending on α) according a first direction, or by a cut edge β (representing all edges depending on β) according a second direction. This can be generalized to any number h of basic cut edges. The fractal nature of the figure seems unavoidable, for copies $\alpha\beta$ and $\beta\alpha$ not to be superimposed, but it is a bit confusing, since in the unfolded graph, all copies are equivalent: there is not a "central copy".

Consider a path in the unfolded graph, starting from a vertex s of some copy, and ending at the same vertex *s but of another copy*. Then this path is a tour in the initial graph: a tour is a circuit not homotopic to ε. The path traverses a sequence of cut edges, labelled $\alpha_1^{\pm 1}\alpha_2^{\pm 1}\ldots\alpha_k^{\pm 1}$. This sequence is called the cut definition of the path; it is obtained just by removing disk edges in the definition of the path (or circuit). The cut definition is sufficient to describe the homotopy of the circuit.

5.3. A method à la Dijkstra

Searching the shortest circuit passing from some vertex s and with some given cut definition $\gamma = \alpha_1^{\pm 1}\alpha_2^{\pm 1}\ldots\alpha_k^{\pm 1}$ reduces to finding the shortest path between a vertex s in some initial copy of the unfolded graph, and the vertex s in the final γ copy. Of course, Dijkstra method can be used for that. The fact that the unfolded graph is infinite is not an insurmountable difficulty: it suffices to generate vertices in a lazy way, only when they are reached. Lazyness, or lazy evaluation, is used in lazy exact arithmetics and geometric computing [BMP94]. Some functional programming languages, like Haskell, are intrinsically lazy: they naturally handle (potentially) infinite data structures, like $\mathbb{N}$ or subsets of $\mathbb{N}$.

After a finite number of steps, the vertex s in the final copy is reached, and the shortest circuit with prescribed homotopy (or cut definition) γ and passing by s is found. Of course, a priori no vertex s of the shortest circuit with homotopy (cut definition) γ is known. But the circuit must pass by some of the n vertices of the initial graph, so we try them all. Actually, it is possible to reduce the number of tries, since the circuit must pass by a relevant vertex: a vertex is relevant when it is an endpoint of some cut edge (an edge the definition of which depends on a basic cut edge). Let α be the basic cut edge which is used in γ (γ uses α or $\overline{\alpha}$), and which has the minimal set A of edges, the cut definition of which depends on α or $\overline{\alpha}$: then the shortest circuit γ must pass by one of the vertices of A. In practice $|A|$ can be an order of magnitude smaller than n.

Unfortunately, this algorithm is not polynomial. It can happen that there is a very short tour, which is not γ: then the Dijkstra method turns around this short tour an arbitrary number (a non polynomially bounded number) of times, before reaching the final vertex. Maybe somebody will find some criteria or theorem which will permit to not to sink deep into an irrelevant very short tour. In absence of such a criteria, this paper proposes another approach.

5.4. Auxilliary points

The cut definition of some edges may contain several basic cut edges. A simple solution adds auxilliary, virtual, points ("intersection point" between the edge and the cuts) on such edges. If the cut definition of the edge (a,b) is $\alpha_1\alpha_2\ldots\alpha_k$ (α_i are basic cut arcs, or their inverses), then $k-1$ auxilliary vertices $a_1,\ldots a_{k-1}$ are created, and edge (a,b) is replaced by edges (a,a_1) having cut definition α_1, (a_1,a_2) having cut definition α_2, … and (a_{k-1},b) having cut definition α_k. The cost of (a,b) is distributed on the new edges. In practice, very few auxilliary points are added, thus there is no consequence on the complexity and running time.

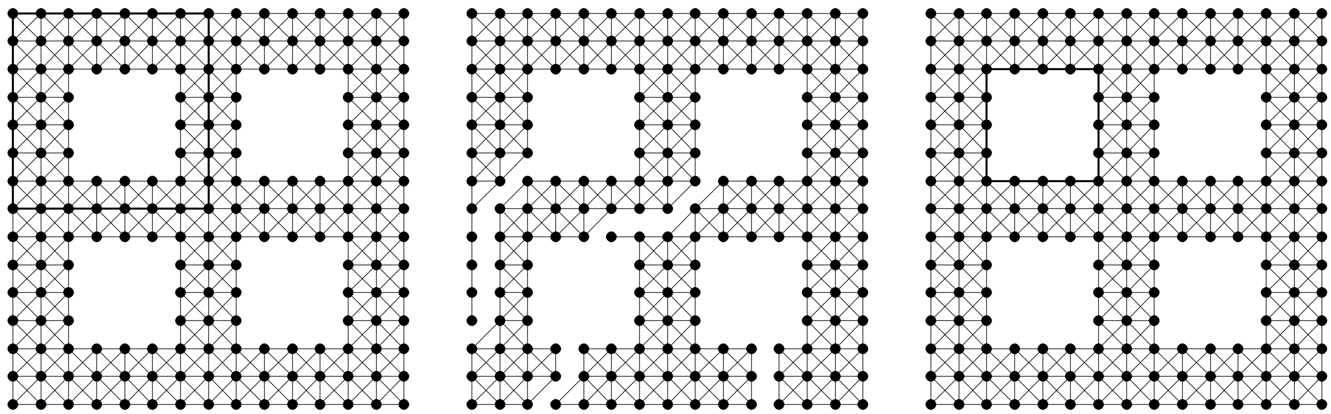

Figure 8: *A trivial example.*

5.5. A three steps method

First, all shortest paths in the unfolded graph between two vertices of the same disk copy are computed (see below, sections 5.6 and 5.8). This first step provides a matrix D, where D_{ij} is the cost of the optimal path from vertex i to vertex j in the unfolded graph. D is symmetric, and its diagonal entries are 0. This cost is smaller or equal to the cost of the optimal path from i to j *in the disk*, and it is greater or equal to the cost of the optimal path from i to j in the original graph.

Second, shortest paths in the unfolded graph between vertices of all pairs of neighboring copies are computed; there is h such pairs of neighboring copies. Let K be the cost matrix of one of these cuts, α: K_{ij} is $+\infty$ if there is no arc ij with cut definition α. Note that, though the initial graph is symmetric, K has no reason to be symmetric: the sets indexed by its rows and its columns are different copies of the disk. For the same reason, K_{ii} is infinite for all i. Then the shortest paths from an initial arbitrary disk copy and its neighbor α is given by the pseudo product DKD (the transpose gives the matrix for $\overline{\alpha}$). The pseudo product $P = AB$ is defined by $P_{ij} = \min_k A_{ik} + B_{kj}$. The analogy of the pseudo product with the usual matrix product is well known, and fast multiplication matrix methods have inspired some fast pseudo products [CLR90]. Due to the complexity of the first step, which is dominant, the naive pseudo product is sufficient. This second step requires $2h$ pseudo products, thus it is $O(hn^3)$. Eventually it provides h matrix for cuts $\alpha_1, \dots \alpha_h$, called $C(\alpha_1)$, $C(\alpha_2) \dots C(\alpha_h)$, and their transposes for cuts $\overline{\alpha_1}, \dots \overline{\alpha_h}$.

Third, for computing the shortest tour with the prescribed homotopy (or cut definition) $\gamma = \alpha_1^{e_1}\alpha_2^{e_2}\dots\alpha_k^{e_k}$ (e_i is ± 1), the matrix M of optimal costs is computed: it is the pseudo product $C_1C_2\dots C_k$, where C_i equals $C(\alpha_i)$ if $e_i = 1$ and the transpose of $C(\alpha_i)$ if $e_i = -1$. M_{ii} is the cost of the optimal tour γ passing by the vertex i: the smallest entry in the M diagonal gives the optimal tour γ. The vertices list of the tour is then rebuilt with one of the usual methods (see [CLR90]). This step is $O(kn^3)$.

5.6. The naive first step method

For short, call a round trip a path or a circuit in the unfolded graph, the starting and ending vertices of which lie in the same disk copy.

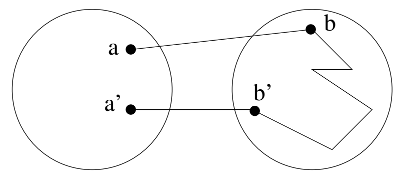

Figure 9: *A shortcut.*

The cut definition of all round trips (optimal or not) is ε. The first step has to compute all optimal round trips.

The simplest method first computes all optimal paths in the disk, either with the Warshall Floyd method [CLR90], or with several calls to the classical Dijkstra method (note the disk is a finite graph). If we are lucky, there is no shortcut (defined below) and optimal round trips are just optimal paths in the disk. A shortcut (see fig. 9) between a vertex a and a vertex a' in the same disk copy is composed of an exit arc ab which belongs to some cut α, an optimal round trip (or the best round trip currently known) from b to b', with b' in the same copy as b, and a return arc $b'a'$ which belongs to the cut $\overline{\alpha}$; finally, to be a shortcut, its cost has to be strictly less than the one of the currently best known path from a to a'. a is the exit vertex, and a' the return vertex, of the shortcut.

The naive method is as follows: each time a shortcut from a to a' is detected ($O(m^2)$ tests are needed, where m is the number of edges in the initial graph, which bounds the number of cut edges), an edge aa' is inserted in the disk, the cost of which is the cost of the found shortcut; if such an edge already exists, its cost is just updated. Possibly, some flag is attached to the edge, to make easier the reconstitution of optimal tours later (the method is inspired by the classical one, presented in [CLR90]). The cost matrix of optimal tours is then updated, either in $O(n^3)$ time with Warshall Floyd method, or in $O(n^2)$ time (section 5.7).

Let l be the maximal number of cut edges appearing in an optimal round trip. Then $l+1$ steps of shortcut detection and updates are needed. Thus this method is $O(n^3 + lm^2n^2)$ time. In practice l is constant: 0, 1, rarely 2. By hand, it is easy to construct (very unlikely) worst cases where $l = O(n)$. We now prove that $l = O(n^2)$: an optimal round trip can not contain two distinct copies D_1 and D_2 with the same exit vertex a and the same return vertex a'; other-

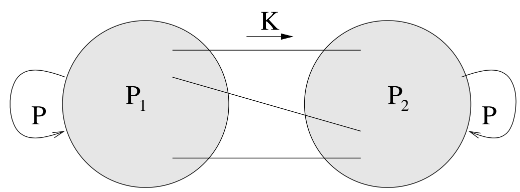

Figure 10: *Two neighboring copies of the disk, linked by a cut α with matrix K. P is the current cost matrix of optimal round trips.*

wise, removing copies between them would give a shortest tour: a contradiction; there are only $O(n^2)$ possible pairs of exit and return vertices; conclude.

5.7. Updating the cost matrix

Updating the cost matrix can be done in $O(n^2)$ time. Let G be the matrix of the non directed graph, with G_{ab} the weight of arc ab ($+\infty$ if there is no arc ab); all diagonal entries are 0. Let D be the cost matrix of optimal paths in G. Let G' be equal to G, except that G' contains one edge uv with cost smaller than D_{uv}. We want to compute the optimal cost matrix D' of G'. D is copied into D'. D'_{uv} and D'_{vu} receive the cost $c_{uv} = c_{vu}$ of uv. All distances to u are updated: for each vertex s, $D'_{su} \leftarrow \min(D'_{su}, D'_{sv} + c_{vu})$ and $D'_{us} \leftarrow D'_{su}$. Symmetrically, all distances to v are updated: for each vertex s, $D'_{sv} \leftarrow \min(D'_{sv}, D'_{su} + c_{uv})$, and $D'_{vs} \leftarrow D'_{sv}$. Then, for each couple s,t in G', the shortest path for st is updated: $D'_{st} \leftarrow \min(D'_{st}, D'_{su} + c_{uv} + D'_{vt}, D'_{sv} + c_{vu} + D'_{ut})$.

5.8. Another first step method

As previously, the optimal paths in the disk are computed. Then successive improvements of round trips are made, until the fix point is reached. Define a *piece* as the complete graph with n vertices, where the cost of each edge ij is the currently known best round trip from i to j (initially it is the optimal path in the disk). Then the graph (fig. 10) with two pieces is considered: the two pieces are linked with cut edges of one of the h cuts. If K is the cost matrix of this cut, then the cost matrix of the graph is:

$$G = \begin{pmatrix} P & K \\ K^t & P \end{pmatrix}$$

where the first line (column) corresponds to the initial copy P_1, the second to its neighbor $P_2 = P_\alpha$. The graph G is symmetric. Optimal paths in G are computed with the Floyd Warshall method. The resulting matrix takes into account for the piece P_1 possible round trips which exit P_1 by some arc α and return by some arc $\overline{\alpha}$, and for the piece P_2 possible round trips which exit P_2 by some arc $\overline{\alpha}$ and return by some arc α. An improvement for P is deduced: $P_{ij} = \min((P_1)_{ij}, (P_2)_{ij})$. An improvement by h cuts calls the Floyd Warshall method h times. Since $l+1$ improvements are needed, this method is $O(n^3 + ln^3)$, which is better than the naive method which is $O(n^3 + lm^2n^2)$.

6. Shortest homologic tour

Figure 11 displays two homologic non homotopic tours (left- and rightmost parts). The two middle parts use one of the possible partitions in disk and cuts, to show that, seen as flows passing through

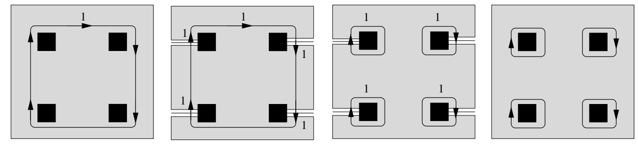

Figure 11: *Left and right tours are homologic.*

the 4 gates (cuts), these two circuits have equal balances at gates: they are homologic, and it does not depend on the chosen partition (or base). Computing the shortest homologic tour can be relevant for some applications, such as optimizing the shape of wirings for a given global electro magnetic field. It is a variant of a max flow min cost network problem. Standard related methods [CLR90, PS98] can be used, at least when the homology matrix is totally unimodular, ie for standard (non exotic) cases. These issues are not discussed here.

7. Conclusion

Constellation are sufficient to perform topologic computations, and are simple and robust. But many questions arise from this first investigation: improve the proposed method; extend it to more general initial surfaces such as surfaces with handles: in this case the unfolded graph of Fig. 6 contains circuits; understand which topologic invariants can be computed from only a constellation; study the shortest homologic circuit problem.

Acknowledgment

Thanks to F. Blais, F. Chazal, E. Colin de Verdière and F. Lazarus, and attendees of the "geodesic day" in Grenoble, June 2003, for enlightening discussions. Thanks to anonymous referees.

References

[BMP94] Benouamer M., Michelucci D., Péroche B.: Error-free boundary evaluation based on a lazy rational arithmetic: a detailed implementation. *Computer-Aided Design 26*, 6 (June 1994), 403–416. 1, 4

[CLR90] Cormen T., Leiserson C., Rivest R.: *Foundations of Computer Science*. MIT Press, Cambridge, Massachusets, 1990. 3, 4, 5, 6

[dVL03] de Verdière E. C., Lazarus F.: Optimal pants decompositions and shortest homotopic cycles on an orientable surface. In *Proc. 11th International Symp. on Graph Drawing (GD'03)* (Sept. 2003). 1

[FK98] Fomenko A., Kunii T.: *Topological Modeling for Visualization*. Springer Verlag, 1998. 2

[Mit97] Mitchell J.: Shortest paths and networks. In *Handbook in Computational Geometry*, Goodman J. E., O'Rourke J., (Eds.). CRC Press, June 1997. 1

[PS98] Papadimitriou C. H., Steiglitz K.: *Combinatorial Optimization*. Dover Publications, 1998. 3, 6

[Veg97] Vegter G.: Computational topology. In *Handbook in Computational Geometry*, Goodman J. E., O'Rourke J., (Eds.). CRC Press, June 1997. 2