CONTINUOUS MAP WITH CONSTRAINTS FOR HOMEOMORPHIC SURFACES

ONG HONG YANG

NATIONAL UNIVERSITY OF SINGAPORE

2004

CONTINUOUS MAP WITH CONSTRAINTS FOR HOMEOMORPHIC SURFACES

ONG HONG YANG

(*B. Eng.* (*Comp. Eng.*) *Hons., NUS*)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2004

ABSTRACT

This thesis investigates the problem of establishing a continuous map between two homeomorphic surfaces, given some correspondence specified over parts of their surfaces. This is the foundation of many morphing techniques with user-specified correspondences. In general, with specified correspondence, it is not always possible to have a continuous map between the two surfaces. This thesis presents an approach that re-expresses each surface as a re-tessellated mesh and systematically analyzes and processes the constraints to get the correspondence over the surface. In particular, it identifies situations where the specified correspondence makes a continuous map between the two surfaces impossible, and computes the necessary relaxation required on the specified correspondence so that a continuous map becomes possible. To stay as close to the user-specified constraints, these modifications are to be kept minimal, and restricted to only the boundary of the regions specified in the constraints.

Keywords: correspondence, morphing, constraint, continuous map, homeomorphism

ACKNOWLEDGEMENTS

My supervisor, Associate Professor Tan Tiow Seng, has been instrumental in making this work possible. On one hand, he has inspired many sparks of insightful thoughts during our discussions as well as provided leads to many related research topics with his wealth of knowledge and experience. On the other hand, he often takes on a constructive opposition role to challenge my theories and ideas, which helped to uncover hidden problems and propel the research in new directions. I would also like to thank him for the great freedom he has allowed me in pursuing a research topic of interest as well as the encouragement when things had seemed disappointing. It has been a great experience and pleasure working with him, and to know him as both a teacher and a friend.

I would like to also thank the many people who have given me advice and help, and all the people whom I have worked with before in research, especially Ms Zhao Yonghong and Mr Xiao Yongguan, with whom I have worked on component-based morphing for a long time. The experience also helped to lay down some of the important foundations on which I started this work.

Last but not least, I would like to thank my family, colleagues and friends who did not help me directly in research, but have been supportive and forgiving towards me when my share of work was delayed or when I went missing for extended periods of time.

CONTENT

LIST OF FIGURES

SUMMARY

This thesis investigates the problem of establishing a continuous map between two homeomorphic surfaces, given some correspondence specified over parts of their surfaces. This is the foundation of many morphing techniques with user-specified correspondences. In general, with constraints, it is not always possible to have a continuous map between the two surfaces. Although much work has been done on morphing and computation of homeomorphisms given some specified correspondence, the issue of conflicts with the goal of getting a continuous map, which may arise from user-specified correspondences, is generally not well addressed.

This thesis presents an approach that re-expresses each surface as a re-tessellated mesh and systematically analyzes and processes the specified correspondences to compute the correspondence over the surfaces by taking into account how the specified correspondences relate to each other. In particular, it identifies situations where the specified correspondence makes a continuous map between the two surfaces impossible, and computes the necessary relaxation required on the specified correspondence so that a continuous map becomes possible. To stay as close to the user-specified constraints, these modifications are to be kept minimal, and restricted to only the boundary of the regions specified in the constraints.

Keywords: correspondence, morphing, constraint, continuous map, homeomorphism

1 INTRODUCTION

The motivation of this work comes from 3-D surface morphing, where one surface-represented object is to transform into another smoothly over time. To do this, the first step is to establish a correspondence between the two spaces representing the two objects, and the second step is to interpolate between the two spaces based on the mapping to generate the intermediate shapes (see survey papers [LV98], [A01], [A02]). Often, some user correspondences are specified between portions of the two surfaces to specify parts of the mapping in order to influence the (appearance of the) transformation. Such user correspondences are known as constraints. As it is a common requirement for morphing to be smooth and continuous, the desired goal is to compute a continuous map between the two surfaces that obeys the given constraints. Subsequently, interpolation techniques are often used to generate the intermediate shapes.

By definition, a continuous map exists between two homeomorphic surfaces. In rubber-sheeting (see [G86], [WG85]), constraints are specified in the form of corresponding pairs of points, one from each of the surface. These constraints represent the warping required to continuously transform one surface to the other. Under any constraints, a continuous map is always possible, though the actual placement of the points determine how slight or severe the warping is.

In many morphing applications, in addition to the actual shape, each point on the surface may also have other attributes, such as color/texture and lighting parameters, that needs to be smoothly transformed as we morph one surface to another. As such, it is usually more natural to specify constraints between areas on the surfaces.

This work extends the component-based morphing approach by Zhao *et al.* [ZOT03]. In that work, the authors address constraints between components where each boundary is shared by exactly two components. Conflict between the specified correspondences and the goal of a continuous map is resolved by adding in additional components that correspond to the original components that have no counterparts. In contrast, this work takes on a fundamental approach in addressing constraints that are general regions on the surfaces that can have boundaries shared by more than two regions, rather than implementing a particular morphing technique. This allows further flexibility in defining morphing, and can be adapted for enhancing the existing

morphing solutions. The main contributions of this work are:

- A systematic approach towards analyzing a set of constraints between regions on the two surfaces in order to identify regions where continuous maps are impossible, and
- A means of computing the necessary relaxation to the given constraints to make a continuous map possible.

Consequently, the original mapping problem can be broken down into much simpler sub-problems, each of which can be solved directly by applying correspondence techniques that uses some standard mapping functions to establish a continuous map between two patches, such as barycentric mapping [ZSH00] and harmonic mapping [KSK00].

The rest of the thesis is organized as follows. Section 2 discusses some of the more related previous works. Section 3 states some preliminary definitions and notations required subsequently, and state the problem statement. The overview of the solution is presented in Section 4, while the subsequent three sections describe the basic solution: Section 5 describes how the user-specified constraints are processed into a standard form; Section 6 analyzes the results from Section 5 to identify the conflicts caused by the constraints and Section 7 computes the necessary changes to resolve these conflicts. Section 8 shows some additional examples that describes how the algorithm works for a special case. Then, Section 9 describes generalization of the basic solution mentioned in Section 5, 6 and 7 to deal with a wider range of problems. Section 10 discusses some of the limitations of current solution as well as future work, followed by the conclusion to this thesis.

2 RELATED WORK

Although there are many related works, especially in the domain of morphing and cartography, this section only mentions the representative works that are relevant, which involve user-specified correspondence (or constraints). Of particular interest is the different ways of dealing with the possibility of conflicts that arises between the constraints and the intended goal of getting a continuous map.

2.1 Computational Geometry and Cartography

The earlier relevant works are those that establish homeomorphisms through compatible triangulations. In these works described in the following subsections, the problem is usually restricted to establishing a continuous mapping between two simple polygons, or to compute compatible triangulations between two regions over two sets of corresponding feature vertices. Many works use only feature points (or feature vertices) to express constraints between the two surfaces. Since no points are specified more than once, there always exist a way to parameterize the two surfaces to get a continuous map. Thus, by choosing to deal only with feature points, conflicts are avoided, rather than solved.

2.1.1 Compatible Triangulation

Aronov *et al.* [ASS93] present a method to compute compatible triangulation between two simple polygons each with n vertices, where the vertices correspond with each other in the same order (clockwise or counter-clockwise). In general, this is not possible, but if additional corresponding vertices (called *Steiner points*) in the polygon interiors can be defined, such a triangulation becomes always achievable.

Souvaine and Wenger [SW94] work with two corresponding point sets lying in the interior of two rectangles in the plane, and construct a piecewise linear homeomorphism. Subsequently, [GW95] and [BSW97] extend the idea to simple polygons and simple polygons with holes.

2.1.2 Rubber Sheet

Rubber sheet is an important problem in cartography addressed by Dey *et al.* [DEG99], the purpose of which is to bring two maps into correspondence: Let M and N be two mappings and $P \subseteq M$ and $Q \subseteq N$ be two sets of same number of points related

by a bijection $b : P \rightarrow Q$. Rubber-sheeting (see [G86], [WG85]) refers to the construction of a homeomorphism $h : M \rightarrow N$ that agrees with b at all points of P. Let $\mathcal{K}$ and $\mathcal{L}$ be simplicial complexes whose simplices cover M and N. Suppose also that the points in P and Q are vertices of $\mathcal{K}$ and $\mathcal{L}$, and that there is a vertex map from vertices of $\mathcal{K}$ to vertices of $\mathcal{L}$ that agrees with b at all points of P. The extension of b to a simplicial map $f : M \rightarrow N$ is a simplicial homeomorphism effectively solving the rubber sheet problem.

2.2 3D Surface Morphing

Morphing is the creation of a smooth series of transitional shapes between two shapes, termed the source and target (for a well-rounded introduction, see survey papers [LV98], [A01], [A02]). 3D Surface morphing is a major sub-category that uses a 3D surface to represent the shape. The morphing process typically consists of two main steps: establishing the correspondence which maps each point on the source to a point on the target, and generating the intermediate shapes, usually by using interpolation.

2.2.1 Using a Base Domain to Deduce Correspondence

DeCarlo and Gallier [DG96] let the user specify correspondences using control meshes over the two surfaces, which divide them into triangular and quadrilateral patches. The control meshes are assumed to be of the same topology except where topological change occurs, which is supported. It is assumed that the constraints specified do not result in conflicts.

Michikawa *et al*. [MKF01] and Praun *et al*. [PSS01] use a common base domain to establish parameterization for a set of meshes. A base domain has to be created, either manually or automatically before hand, and all the corresponding vertices to those in the base domain have to be specified over the meshes. The surfaces are partitioned consistently by this base domain. Subsequently, remeshes, which are new meshes built by sampling the original meshes, are created over the surfaces.

2.2.2 Aligning Features by Warping

Lee *et al*. [LDS99] support the specification of constraints using both vertices and edges, and use them to create base domains for the two surfaces to be corresponded. The base domains are aligned on the feature vertices and edges before they are merged

to form a correspondence map. The correspondence is then propagated to the rest of the original surfaces.

Alexa [A00] allows scattered feature vertices to be specified, and employs spherical mapping on the surfaces, and perform warping to align the feature vertices. The two spheres are then merged to create a common mesh.

2.2.3 Partitioning into Corresponding Patches

Gregory *et al*. [GSL98] construct a feature net on each surface from feature vertices specified by the user, which partitions the surface into patches. A bijection between the two feature nets is assured by imposing some restrictions on the user on the way the vertices and the chains joining the feature vertices are specified, which avoids the issue of conflicts.

Kanai *et al*. [KSK00] construct control meshes that partition the surfaces from given vertex correspondences. It differs from [GSL98] in that it uses geodesic curves and harmonic mapping instead of vertex-edge paths and area-preserving mapping employed by the latter.

Zöckler *et al*. [ZSH00] allow the specification of feature patches in addition to feature vertices, and decompose the two surfaces into two topologically equivalent sets of patches, before barycentric mapping to align and merge the patches. It is assumed that the feature patches specified do not result in conflicts.

Zhao *et al*. [ZOT03] present a hybrid framework that partitions each surface into a set of *components*. Correspondences can be specified at two levels: between the components at a high-level, and in the form of feature vertices and feature edges within a pair of corresponding components at the low-level. Correspondences are specified between two sets of components and are not assumed to produce naturally compatible outcome. Instead, the user-specified correspondences are processed to detect any conflicts, in the form of components of one surface not having correspondents in the other surface. Then, the conflicts are resolved by adding dummy components to make the two sets of corresponding components topologically compatible. This is made easier by the assumption that a boundary between two components is shared exactly by two components. Correspondences specified by feature vertices and feature edges within a pair of corresponding components is similar

to the common problem of having scattered features within a pair of corresponding patches and is handled by an adaptation of techniques in the other works.

2.3 Summary

Among the works listed, only a few actually deal with the situation where user-specified correspondences make it impossible to get a continuous map between the two surfaces being corresponded. Some simply assume that their input does not contain such a case, thus avoiding the problem totally. Others adopt frameworks that place certain restrictions on the user, e.g. by assuming that the two surfaces have topologically compatible base domains, which are assumed to be properly specified by human. The most commonly seen approach is to only deal with scattered features (feature vertices or edges) on two homeomorphic surfaces, on which a continuous map that obeys the given constraints is always possible. This work aims at giving the user as much freedom as possible in the specified correspondences, and tries to satisfy all these specifications, only changing them when necessary in getting a continuous map.

Zhao *et al*. [ZOT03] presents certain ideas which are closest to this work in spirit: the user-specified correspondences are processed systematically to work out additional implications of the given correspondences, as well as detecting conflicts that arises between the given correspondences and the goal of getting a continuous map. The actual solution differs from this thesis in many areas. Their work simplifies the problem by assuming that boundaries between components are shared exactly between two components. Conflicts between the specified correspondences and the goal of a continuous map are resolved by adding in additional components that correspond to the original components that have no counterparts. Our method works with general patches as constraints, which have boundaries shared by more than two patches. Conflicts are resolved by relaxing the constraints near the boundaries of the regions specified in the constraints, rather than adding in new patches. In summary, our work should not be seen as an independent morphing technique, but rather an algorithm that analyzes the constraints, detects the conflicts with the goal of achieving a continuous map, and produces a mapping that is continuous between the two surfaces by computing the necessary changes that need to be made on the given constraints.

3 PROBLEM STATEMENT

This section states the problem as well as introduces some basic notations used in the rest of the thesis.

3.1 Basic Definitions

Let X and X' be the two genus 0 manifolds, on which we want to construct a continuous map, and are represented as polygonal meshes of the same genus. Since we are only interested in the topology, X and X' can be interpreted as simplicial complexes. In fact, the mesh representation is chosen to simplify the problem over an arbitrary surface where it is assumed that all the constraints can be specified over an integral number of polygonal faces. In practice, the mesh can be dynamically triangulated to accommodate any free-form constraints specified over the surfaces.

The sets of faces, edges and vertices of a simplicial complex X are denoted as *faces*(X), *edges*(X), and *vertices*(X), respectively. A *region* F in X is a subset of X. In addition, for all $e \in$ *edges*(X), we have $e \in F$ if both incident faces of e are in F; for all $v \in$ *vertices*(X), we have $v \in F$ if all incident edges of v are in F. The *boundary* of $F \subseteq X$ is *boundary*(F) = *cl*(F) $\cap$ *cl*($X - F$) where *cl* is the closure function. A *correspondence* between a region F in X and a region F' in X' is denoted as $\langle F, F' \rangle$. Each $\langle F, F' \rangle$ denotes that F is to be bijectively mapped to F' only via a function φ, where $\varphi(F) = F'$ and $\varphi^{-1}(F') = F$. A *constraint* is a correspondence which is specified by the user. The set of constraints is denoted as

$$\mathcal{S} = \{ \langle F_i, F'_i \rangle : F_i \subseteq X, F'_i \subseteq X', i = 1, 2, \ldots, n \}$$

From here till Section 8, it is assumed that for all $\langle F_i, F'_i \rangle \in \mathcal{S}$, both F_i and F'_i are topologically disks, and for any $\langle F_i, F'_i \rangle$, $\langle F_j, F'_j \rangle \in \mathcal{S}$, neither $F_i - F_j$ nor $F'_i - F'_j$ results in more than one connected subset of X and X'.

3.2 Objective

Given two homeomorphic manifolds X and X' and a set of constraints $\mathcal{S}$, find a bijective mapping between X and X':

$\{\langle G_j, \hat{G_j}\rangle : \bigcup_j \{G_j\}$ and $\bigcup_j \{G'_j\}$ are partitions of X and X', respectively$\}$,

where the constraints are obeyed and continuity is preserved. That is, for every $\langle G_j, \hat{G_j}\rangle$, we have for all $\langle F_i, \hat{F_i}\rangle \in \mathcal{S}$, $G_j \subseteq F_i \leftrightarrow \hat{G'_j} \subseteq \hat{F'_i}$. And that for any $\langle G_i, \hat{G_i}\rangle$ and $\langle G_j, \hat{G_j}\rangle$ where $i \neq j$, G_i and G_j are incident if and only if $\hat{G'_i}$ and $\hat{G'_j}$ are incident. A *conflict* is said to occur when a mapping which satisfies all the conditions is not possible. If this is the case, in order to find a continuous map, some constraints in $\mathcal{S}$ have to be *relaxed*.

This is done by modifying the constraints to exclude small buffer regions at their boundaries, in order to resolve problematic areas that cannot be continuous under the original constraints. We want to minimize this relaxation.

4 APPROACH

This section first describes the basic idea behind the relaxation of constraints by using re-tessellated meshes (Section 4.1). Then, the original problem is expressed under this new meshes (Section 4.2). Subsequently, conflict under this representation is discussed (Section 4.3), and how the relaxation works is explained (Section 4.4).

4.1 Surface Interpretation

To relax between two constraints, the local space near their common boundaries can be treated as a region that is not part of both constraints, as illustrated in Figure 4-1. Thus, certain singularity issues arising from them can be resolved, by manipulating the correspondence of this new buffer region. Implementation-wise, a simple way is to re-tessellate the space near certain subsets of edges and vertices in X and X' involved in the relaxation, so that they become narrow faces, which can be excluded from the constraints specified on the original meshes.

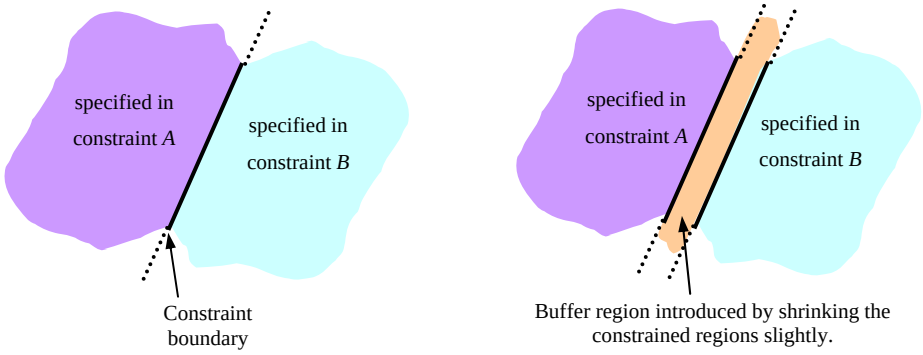


Figure 4-1: Constraint relaxation

To cater to all possible relaxations, we pre-process all the edges and vertices in X and X' to be new faces, such that all possible relaxations can be represented in a discrete manner on these new representations, call *re-tessellated meshes*. In the process, we convert each of the simplices in the original mesh to a polygon as follows (see Figure 4-2).

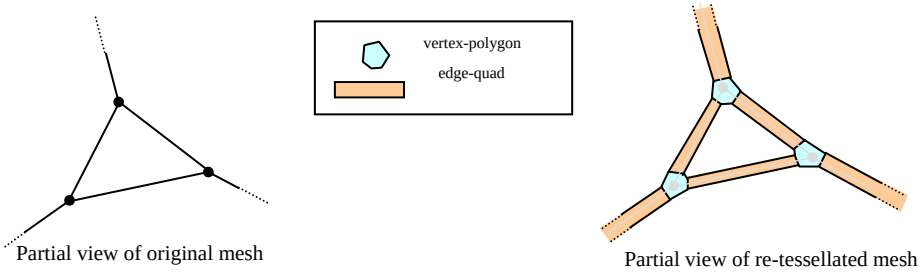


Figure 4-2: Mesh Re-tessellation

- Each vertex of degree *n* in the original mesh becomes a *vertex-polygon* of 2*n* sides in the re-tessellated mesh, where the 2*n* sides are shared with *n* quads, which were the incident edges in the original mesh (see Figure 4-3).

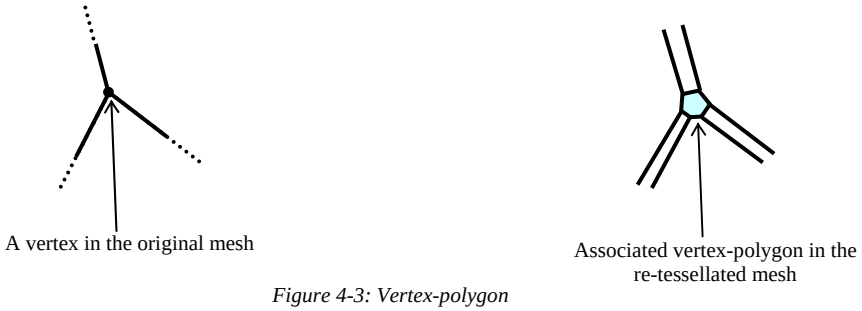


Figure 4-3: Vertex-polygon

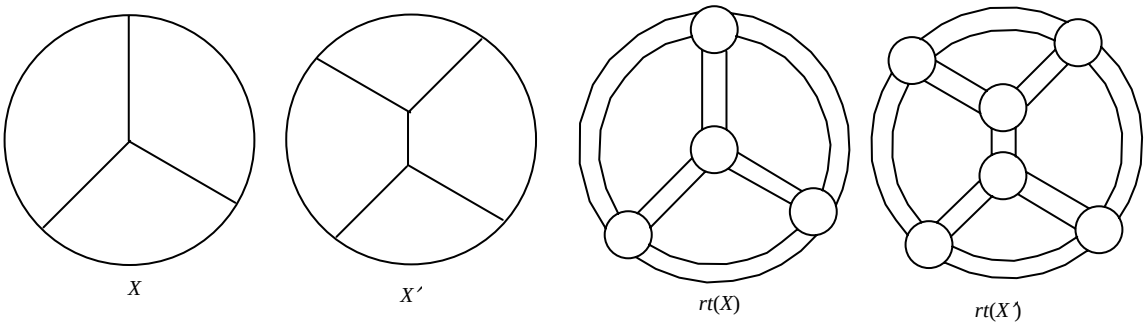
- Each edge in the original mesh becomes an *edge-quad* in the re-tessellated mesh, whose sides are shared with two vertex-polygons and two incident faces (see Figure 4-4).

Figure 4-4: Edge-quad

- Each polygonal face of *n* sides in the original mesh becomes a *face-polygon* that has 2*n* sides shared with *n* vertex-polygons and *n* edge-quads in the re-tessellated mesh (see Figure 4-5).

Figure 4-5: Face-polygon

This re-tessellated mesh is a new mesh where the degree of each vertex is exactly 3; see Figure 4-6. This means that there are no cases where the closures of any two different polygons intersect at only a vertex, which makes subsequent processing much easier.



(a) This figure shows the structure of the mesh used to represent two surfaces *X* and *X′*.

(b) This figure shows how the surfaces are represented under *rt(X)* and *rt(X′)*.

Figure 4-6: Example: Mesh re-tessellation

4.2 Problem Revisit

In a re-tessellated mesh, its vertices and edges are not of significance as it is the incidence relationship among faces that are of interest. Thus, unlike the original mesh, which is represented by a simplicial complex, a re-tessellated mesh is a set of faces, comprising vertex-polygons, edge-quads and face-polygons. Let *rt* be the bijective function that maps each simplex in a mesh *X* to its corresponding face in the re-tessellated mesh of *X*, denoted $rt(X) = \{rt(x) : x \in X\}$. The function rt^{-1} is used to denote the inverse mapping of *rt*. An *rt-region R* is defined to be a connected subset $R \subseteq rt(X)$ where $rt^{-1}(R)$ is a connected subset of *X*. A constraint $\langle F, F' \rangle$ between *X* and *X′* can be expressed as a constraint $\langle rt(F), rt(F') \rangle$ with respect to the re-tessellated surfaces $rt(X)$ and $rt(X')$. Accordingly, the set of all constraints can be represented as $rt(\mathcal{S}) = \{\langle rt(F), rt(F') \rangle : \langle F, F' \rangle \in \mathcal{S}\}$.

Two *rt*-regions $R_1 \subseteq rt(X)$ and $R_2 \subseteq rt(X)$ are said to be *adjacent* if and only if $rt^{-1}(R_1)$ and $rt^{-1}(R_2)$ are incident to each other in the original mesh. A *neighbor* of an *rt*-region *R* is an *rt*-region that is adjacent to *R*. A *neighbor ring* of an *rt*-region *R* is the list of all neighbors of *R* in a counter-clockwise order around *R* and is denoted as $N(R)$. Neighbor rings are cyclically invariant.

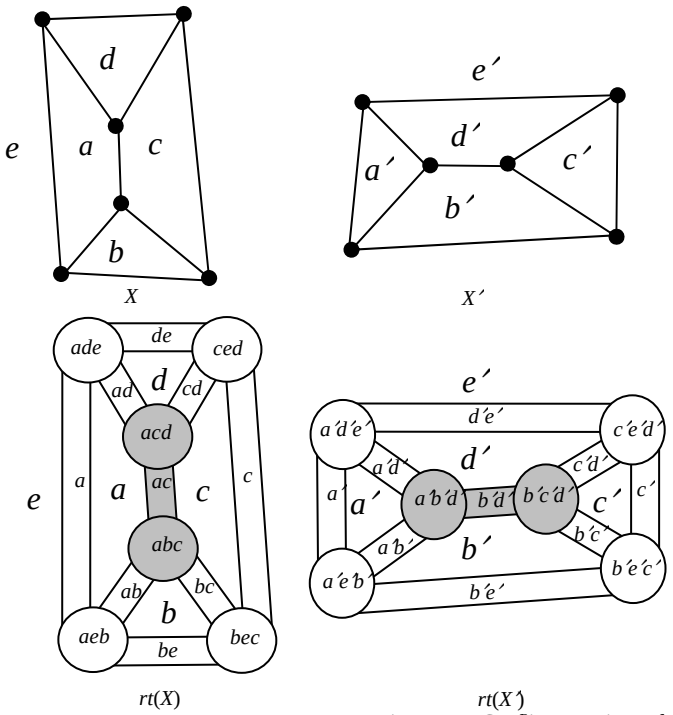
We define the following: two *rt*-regions $R \subseteq rt(X)$ and $R' \subseteq rt(X')$ are *correspond-able* if and only if they are included or excluded by the same set of constraints in $rt(\mathcal{S})$, i.e., $\forall \langle A, A' \rangle \in rt(\mathcal{S}),\ (R \subseteq A) \leftrightarrow (R' \subseteq A')$. Two neighbor rings *correspond* if there is some way of laying out all the regions in the two neighbor rings in pairs, while maintaining a

counter-clockwise order, such that the regions in each pair are correspond-able.

At this point, we re-state the problem as follows. Given two homeomorphic surfaces $rt(X)$, $rt(X')$ and a set of constraints $rt(\mathcal{S})$, find a mapping $\{\langle R_i,R'_i\rangle : i = 1, 2, \ldots, n\}$, where $\{R_1, R_2,..., R_n\}$ forms a partition of $rt(X)$ and $\{R'_1, R'_2,..., R'_n\}$ forms a partition of $rt(X')$ such that for each $\langle R_i,R'_i\rangle$, R_i and R'_i are correspond-able and the neighbor rings of R_i and R'_i correspond.

4.3 Conflict Detection

A *conflict* occurs when for some $\langle A, A'\rangle \in rt(\mathcal{S})$, the neighbor rings of A and A' do not correspond. Consider the example of Figure 4-7(a). It shows a particular region of the object X and X', related by the constraint set $\mathcal{S} = \{\langle a,a'\rangle, \langle b,b'\rangle, \langle c,c'\rangle, \langle d,d'\rangle, \langle e,e'\rangle\}$.



(a) The diagram of the left shows the regions specified in $\mathcal{S}$.

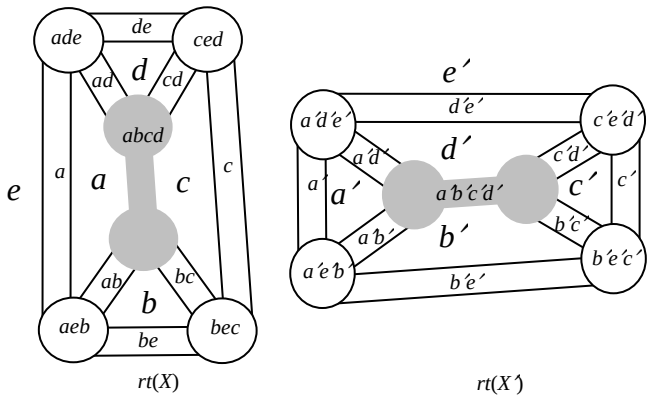
(b) The diagram of the left shows the representation of diagram (a) in $rt(X)$ and $rt(X')$. The edge-quads and vertex-polygons are identified by the incident faces.

Figure 4-7: Conflict at regions of discontinuity

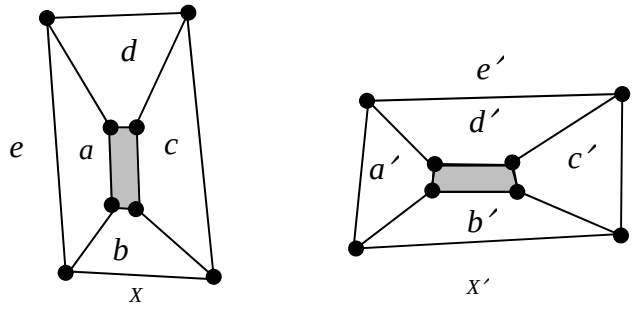
In this case, there is no continuous map between X and X' as the constraint set cannot be resolved in a satisfactory manner: In X, regions a and c are adjacent and regions b and d are not, but in X' regions a' and c' are not adjacent and regions b' and d' are. This introduces conflicts, shown as shaded in Figure 4-7, are in the neighbor rings of some of the *rt*-regions in $rt(X)$ and $rt(X')$. In other words, any mapping that obeys the given constraint set exactly is not continuous around the shaded *rt*-regions.

4.4 Constraint Relaxation

Conflicts are resolved by relaxing the constraints around the boundaries of the specified regions. The basic idea is to remove the distinction between certain adjacent *rt*-regions and treat them as a single homogenuous region. This is termed as *merging* of *rt*-regions. The resultant *rt*-region has an updated neighbor rings. During the relaxation process, merging is carried out if at least one of the *rt*-regions to be merged does not contain face-polygons specified in some constraints. This is a way of ensuring that any specified constraints on face-polygons are obeyed. A solution for the conflict shown in Figure 4-7 is shown below in Figure 4-8, by merging the shaded *rt*-regions in each surface such that the resultant *rt*-regions have corresponding neighbor rings and can be made corresponding. All their neighbors' neighbor rings then become corresponding as well.



(a) From the example in Figure 4-7(b), merging can be done on the shaded regions..



(b) The diagram shows what the merging in (a) represents on the original surface. The modification of the boundaries of the constraint regions creates a very narrow buffer, which is free from all the original constraints. A continuous map becomes possible, after making the two buffer regions correspond to each other.

Figure 4-8: Relaxing constraints through merging of rt-regions.

In general, relaxation should be kept to the minimum and only done when necessary, to ensure that user-specified constraints are adhered to as much as possible. This is the guiding philosophy throughout this work. As such, merging is never done on face-polygons specified in constraints.

If the specified constraints are composed entirely of pairs of regions which are topologically disks, the trivial solution to get a continuous map can be achieved by merging all the constraint boundaries and any regions not specified in any constraints.

This results in the basic case where all the regions specified in constraints are independent, and a continuous map is always possible.

4.5 Solution Overview

The solution presented in this report have three major phases:

- Constraint Processing. (Section 5) In the first phase, the set of general user-specified constraints are pre-processed to represent simple constraints on the re-tessellated surfaces $rt(X)$ and $rt(X')$.

- Conflict Analysis. (Section 6) In the second phase, means of detecting conflicts are set up. Simple cases of conflicts, which are unambiguous, are resolved, before the necessary relaxations on the constraints are done in the next phase.

- Constraint Relaxation. (Section 7) In this phase, the desired situation is that the boundary regions around all faces corresponded by the constraints must be continuously mapped. Where this is impossible, the constraint is relaxed around the boundaries. The aggregated result is a continuous map across the two surfaces.

The meshes in Figure 4-6 is used as an example throughout Sections 5 and 6. Additional examples are shown in Sections 7 and 8.

5 CONSTRAINTS PROCESSING

In the first phase, the set of constraints are processed to represent constraints on the re-tessellated surfaces $rt(X)$ and $rt(X')$. In particular, it computes (in Sections 5.2 and 5.3) a complete correspondence set $\mathcal{C}$, which is a simple set of correspondence over the entire surfaces $rt(X)$ and $rt(X')$. Each element of $\mathcal{C}$ captures a pair of regions that are homogeneous in their inclusion in a subset of the constraint $\mathcal{S}$. Section 5.4 shows an example of this phase.

5.1 Overview

When we specify some constraint $\langle F_i, \hat{F_i} \rangle \in \mathcal{S}$, it is reasonable to assume that their boundaries will correspond to each other as well. To be able to derive the correspondence of the edge-quads and vertex-polygons, each constraint $\langle F_i, \hat{F_i} \rangle \in \mathcal{S}$ are said to induce correspondences over all edges and vertices on the boundaries of F_i and $\hat{F'_i}$, i.e., $\langle F_i, \hat{F_i} \rangle \in \mathcal{S}$ implies the correspondence $\langle rt(cl(F_i)), rt(cl(\hat{F_i})) \rangle$ on $rt(X)$ and $rt(X')$.

To record the correspondence over all of $rt(X)$ and $rt(X')$, we define *complete correspondence set* $\mathcal{C}$ to be a set $\{ \langle Id_j, B_j, \hat{B_j} \rangle : j = 1, 2, ..., n \}$, where $\{B_1, B_2, ..., B_n\}$ forms a partition of $rt(X)$ and $\{\hat{B'_1}, \hat{B'_2}, ..., \hat{B'_n}\}$ forms a partition of $rt(X')$, and there is a bijection between every $y \in B_j$ to some $y' \in \hat{B'_j}$. Id_j is a set of identifiers, where every $i \in Id_j$ implies that $rt^{-1}(B_j) \subseteq cl(F_i)$ and $rt^{-1}(\hat{B_j}) \subseteq cl(\hat{F_i})$ where $\langle F_i, \hat{F_i} \rangle \in \mathcal{S}$. In other words, for all $i \in Id_j$, the regions $rt^{-1}(B_j)$ and $rt^{-1}(\hat{B_j})$ are included in the ith constraint in $\mathcal{S}$,.

5.2 Constructing Complete Correspondence $\mathcal{C}$

To eventually incorporate the correspondence induced by a constraints $\langle F_i, \hat{F_i} \rangle \in \mathcal{S}$ on the boundaries of F_i and $\hat{F'_i}$, we first define $\mathcal{H}$ to contain the set of constraints as follows:

$\forall \langle F_i, F'_i \rangle \in \mathcal{S},\ \langle Id_i, A_i, A'_i \rangle \in \mathcal{H}$

where

$A_i = rt(\ cl(F_i)\ )$ is the *rt*-region induced by F_i ,

$A'_i = rt(\ cl(F'_i)\ )$ is the *rt*-region induced by F'_i, and

$Id_i = \{i\}$ is an identifier set used to uniquely identify a constraint $\langle F_i, F'_i \rangle \in \mathcal{S}$.

$\mathcal{C}$ is initialized to $\{\ \langle \varnothing, rt(X), rt(X') \rangle\ \}$, which is the trivial correspondence between the two original surfaces before any constraints are added. $\varnothing$ as the identifier set indicates that initially, there are no constraints. Next, elements in $\mathcal{H}$ are incorporated into $\mathcal{C}$ one by one as follows: For each $\langle Id_i, A_i, A'_i \rangle \in \mathcal{H}$, go through every $\langle Id_j, B_j, B'_j \rangle \in \mathcal{C}$ to replace it with:

$\langle Id_i \cup Id_j, A_i \cap B_j, A'_i \cap B'_j \rangle$ which is the intersection, and

$\langle Id_j, B_j - A_i \cap B_j, B'_j - A'_i \cap B'_j \rangle$ which is the remainder.

For $\langle Id_j, B_j, B'_j \rangle \in \mathcal{C}$, the identifier sets of B_j and B'_j are denoted as $id(B_j)$ and $id(B'_j)$ respectively and are equal to Id_j.

See Figure 5-1 for an example.

Partial View of X Partial View of X′

In the figure above, three constraints $S = \{ \langle a, a' \rangle, \langle b, b' \rangle, \langle c, c' \rangle \}$ are specified. Thus, $\mathcal{H} = \{ \langle \{1\}, a, a' \rangle, \langle \{2\}, b, b' \rangle, \langle \{3\}, c, c' \rangle \}$. The resultant identifiers for each rt-region are shown above in the braces.

Figure 5-1: Constraint Boundaries

Each resulting tuple of the form $\langle Id, \varnothing, \varnothing \rangle$ for some identifier set *Id*, is not relevant and can be discarded from $\mathcal{C}$. Any tuple with $\varnothing$ as the second or third parameter, such as $\langle Id_j, B_j, \varnothing \rangle$ or $\langle Id_j, \varnothing, B'_j \rangle$, is called an *unmatched constraint*. Non-null B_j or B'_j in such a tuple is called an *unmatched region*, which is an *rt*-region in one surface for which a

corresponding *rt*-region cannot be found in the other surface. Any tuple with ∅ as the identifier set, such as $\langle \varnothing, B_j, B'_j \rangle$, is called an *unspecified constraint*. The *rt*-regions B_j and B'_j in these tuples, called *unspecified regions*, denote parts of the surface where the correspondence is not specified in constraints, and thus unknown.

5.3 Processing Unspecified Constraints

The idea is to assign a temporary constraint to each unspecified region so that itself, and its adjacent vertex-polygons and edge-quads can be detached from other constraints to become unmatched regions. Such region will then go through subsequent processing to inherit all the identifiers of the neighboring vertex-polygons and edge-quads, and from there to infer its relative positioning with respect to other regions and thus its correspondence.

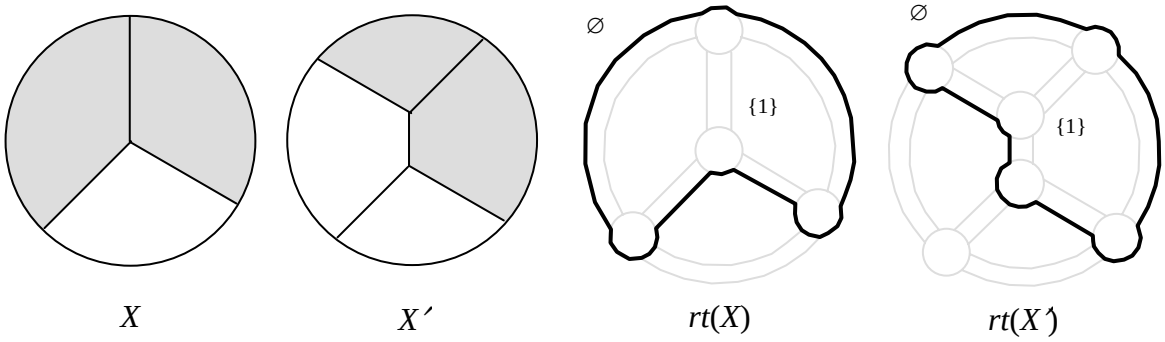
Specifically, we locate the tuple $\langle \varnothing, B, B' \rangle \in \mathcal{C}$, where B and B' are excluded from all constraints $\langle Id_i, A_i, A'_i \rangle \in \mathcal{H}$. This tuple exists unless the constraints in $\mathcal{H}$ cover the whole of $rt(X)$ and $rt(X')$, in which case there are no unspecified constraints. If there are more than one maximally connected subset in $rt^{-1}(B)$ and $rt^{-1}(B')$, partition $\langle \varnothing, B, B' \rangle$ into smaller tuples $\{\langle \varnothing, B_i, \varnothing \rangle : i=1, 2, \ldots, m\}$ and $\{\langle \varnothing, \varnothing, B'_j \rangle : j = 1, 2, \ldots, m'\}$ such that $\{B_1, B_2, \ldots, B_m\}$ forms a partition of B and $\{B'_1, B'_2, \ldots, B'_m\}$ of B', and each $rt^{-1}(B_i)$ and $rt^{-1}(B'_j)$ is one maximally connected subset of X and X' respectively.

A temporary identifier set *Tid* is used to store the identifiers of the temporary constraints defined on the unspecified regions. It is initialized to ∅. Then, go through each $\langle \varnothing, B_i, \varnothing \rangle \in \mathcal{C}$, and let $Id_{temp} = \{m\}$ where m is a new unique identifier not previously used (negative integers are used in the implementation). For each case, create a new constraint over B_i and its neighboring vertex-polygons and edge-quads as $\langle Id_{temp}, rt(cl(rt^{-1}(B_i))), \varnothing \rangle$ and proceed to incorporate it into $\mathcal{C}$ in the manner described in Section 5.2. Record the identifier by making $Tid = Tid \cup Id_{temp}$. Similarly, for each case of $\langle \varnothing, \varnothing, B'_i \rangle \in \mathcal{C}$, a temporary constraint is defined over B'_i and its neighboring vertex-polygons and edge-quads as $\langle Id_{temp}, \varnothing, rt(cl(rt^{-1}(B'_i))) \rangle$ to be incorporated into $\mathcal{C}$.

5.4 Example

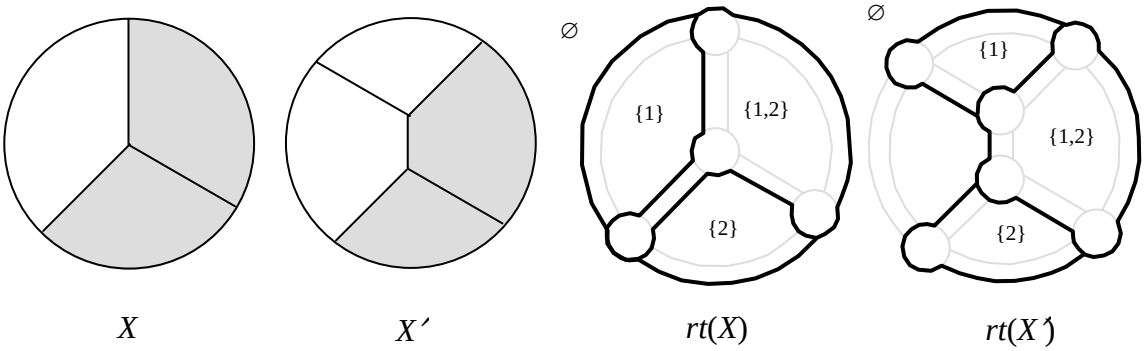
This example continues from Figure 4-6. First, The complete correspondence $\mathcal{C}$ is created over the whole re-tessellated surfaces in Figure 4-6(b). In this example, three constraints are added, which are indicated as shaded regions in Figure 5-2(a), (c) and (e). The resultant partitioning of the re-tessellated meshes indicated by $\mathcal{C}$ is shown in Figure 5-2(b), (d) and (f) respectively. There is one unspecified region indicated by $\varnothing$ in $rt(X')$ in Figure 5-2(f). After a temporary constraint is defined over the unspecified region, the result is shown in Figure 5-2(g), where the unmatched regions caused by the temporary identifier is colored gray.

Figure 5-2: Example: Constraint Processing



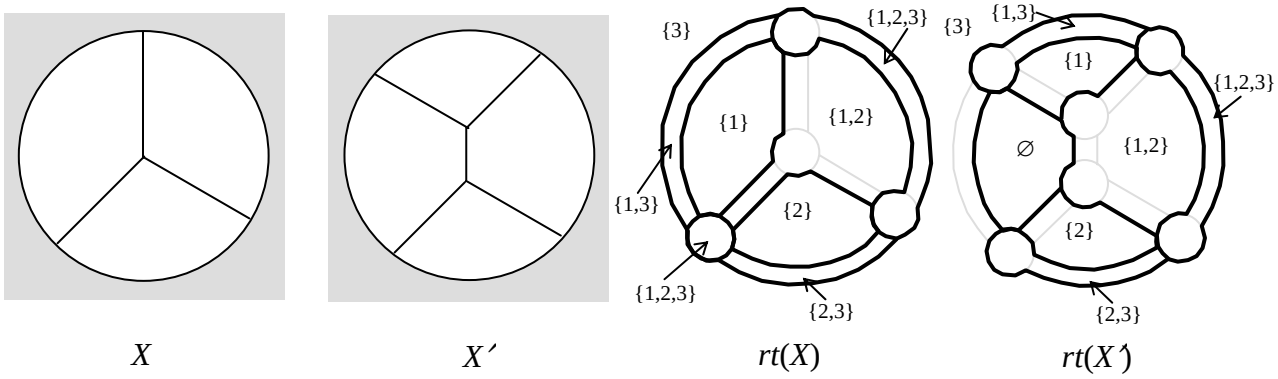
(a) The shaded regions in this diagram represent the correspondence specified by a constraint.

(b) This shows the corresponding changes in complete correspondence $\mathcal{C}$, and the identifier set for every region. The bold lines indicate how the surfaces are partitioned at the moment in $\mathcal{C}$.



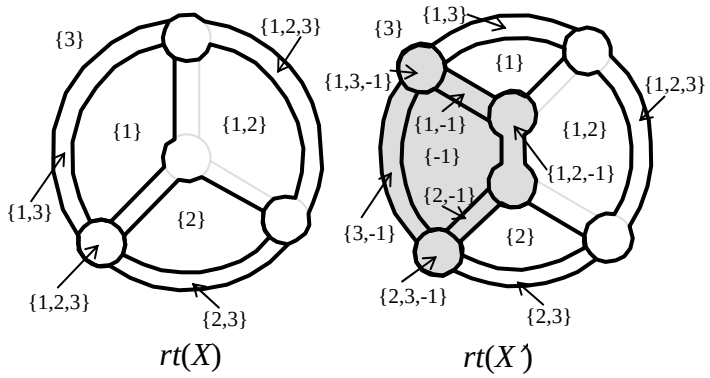
(c) The shaded regions in this diagram represent the correspondence specified by a second constraint.

(d) This shows the corresponding changes in complete correspondence $\mathcal{C}$ after second constraint. The bold lines indicate how the surfaces are partitioned at the moment in $\mathcal{C}$.



(e) The shaded regions in this diagram represent the correspondence specified by a third constraint. This is the last constraint.

(f) This shows the corresponding changes in complete correspondence $\mathcal{C}$ after the third constraint. The bold lines indicate how the surfaces are partitioned at the moment in $\mathcal{C}$.



(g) At this point, there are no further constraints. The unspecified regions are processed as if they are constraints but with a temporary identifier (-1 in this case). The unmatched regions are shown in grey. Constraint processing is done.

6 CONFLICT ANALYSIS

This section illustrates how the complete correspondence set $\mathcal{C}$ resulting from Section 5 is further processed to prepare it for the final computation of the relaxations required of the constraints. Section 6.1 gives an overview of the process at this step and define certain terms and concepts. Section 6.2 describes the process of inferring the correspondence for unmatched regions from known correspondence adjacent regions.

6.1 Overview

At this stage, the complete correspondence set $\mathcal{C}$ contains the basic correspondence information over the entire surfaces. Every part of a surface is either already under a constraint which is already locally continuous, or under a constraint which not locally continuous, or has no known correspondence. To check whether an *rt*-region is continuous, the *neighbor ring identifier list* is first defined: for a neighbor ring of an *rt*-region $B \in rt(X)$ where $N(B) = (B_1, B_2, \ldots, B_x)$, the neighbor ring identifier list of B, denoted $Nrid(B)$, is defined as follows:

$Nrid(B) = id(N(B)) = (id(B_1), id(B_2), \ldots, id(B_x))$ and is cyclically invariant. For any *rt*-regions B and B', $Nrid(B) = Nrid(B')$, if, and only if $id(N(B))$ and $id(N(B'))$ are cyclic permutations of each other. The tuples in $\mathcal{C}$ can be classified as follows:

- $\langle Id_i, B_i, \hat{B_i} \rangle \in \mathcal{C}$, where $Nrid(B_i) = Nrid(\hat{B_i})$.

  This denotes constraints whose *rt*-regions can already be mapped bijectively, as the immediate neighbors already correspond.

- $\langle Id_i, B_i, \hat{B_i} \rangle \in \mathcal{C}$, where $Nrid(B_i) \neq Nrid(\hat{B_i})$.

  This denotes constraints whose *rt*-regions, which are supposed to be corresponding, have different neighbor rings identifier list. Merging has to be done to relax the constraints around the border.

- $\langle Id_i, B_i, \varnothing \rangle \in \mathcal{C}$ or $\langle Id_i, \varnothing, \hat{B_i} \rangle \in \mathcal{C}$

  This denotes unmatched regions whose correspondence are unknown. The correspondence has to be derived from the correspondence of their neighbors.

This stage focuses on the last group consisting of *rt*-regions with no known correspondence. Their correspondence are inferred from the correspondence of other regions in their immediate neighborhoods.

6.1.1 Neighborhood Difference

A *neighborhood* of a face-polygon is defined to be a circular-invariant list of all the adjacent vertex-polygons in the counterclockwise direction. Let $A \in rt(X)$ and $B \in rt(X')$ be two corresponding face-polygons with the neighborhoods $(A_1, A_2, \ldots, A_n)$ and $(B_1, B_2, \ldots, B_m)$ respectively. The n*eighborhood difference* is a quantitative measure of how well the neighborhoods of A and B corresponds on the whole.

To compute the neighborhood difference, we try to *align* the elements in the neighborhoods $(A_1, A_2, \ldots, A_n)$ and $(B_1, B_2, \ldots, B_m)$ as much as possible. Initially, two elements A_k, $k \in \{1, 2, .., n\}$ and B_z, $z \in \{1, 2, .., m\}$ where $id(A_k) = id(B_z)$ are aligned. The circular lists representing the neighborhoods are rotated such that A_k and B_z appear as the first element of each list. Subsequently, for the other elements, some A_i, $i \in \{1, 2, .., n\}$ and some B_x, $x \in \{1, 2, .., m\}$ can be *aligned* if the following are satisfied:

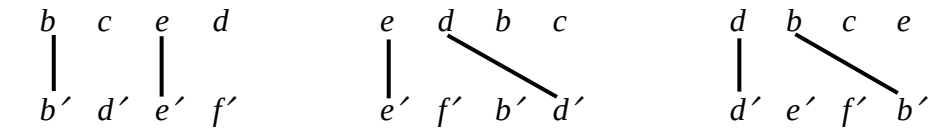
1. A_i and B_x are corresponding, i.e. $id(A_i) = id(B_x)$, and

2. For every other pair of existing aligned elements in the neighborhoods A_j and B_y, where $A_j \neq A_i$ and $B_y \neq B_x$, then A_j appears before A_i in the current listing order of the neighborhood of $A \in rt(X)$ if and only if B_y appears before B_x in the current listing order of the neighborhood of $B \in rt(X)$.

For a pair of aligned neighborhoods with a set of pairs of aligned elements, it is possible to rotate the circular lists representing the neighborhoods such that one pair of aligned elements appear as the first element in each list. The *neighborhood difference* function between neighborhoods of A and B is denoted as $Ndiff(A, B)$ and gives the number of unaligned elements in the case of the properly aligned neighborhoods with the largest number of pairs of aligned elements.

For example, assuming a and a' is a pair of corresponding face-polygons with the neighborhoods (b,c,e,d) and (b', d',e',f') respectively, and $\{ \langle Id_2,b,b' \rangle, \langle Id_3,c,\varnothing \rangle, \langle Id_4,d,d' \rangle, \langle Id_5,e,e' \rangle, \langle Id_6,\varnothing,f' \rangle \} \in C$, $Nrid(a) = (Id_2, Id_3, Id_5, Id_4)$ and $Nrid(a') = (Id_2,$

Id_4, Id_5, Id_6). Figure 6-1 shows three possible ways to align the elements. The neighborhoods of a and a' are displayed in two rows and the aligned elements in each case are connected by straight lines. In all the cases shown, the number of unaligned elements is 4, which is already the lowest possible. Thus, $Ndiff(a, a') = 4$.



set of unaligned elements = $\{c, d, d', f'\}$ | set of unaligned elements = $\{b, c, f', b'\}$ | set of unaligned elements = $\{c, e, e', f'\}$

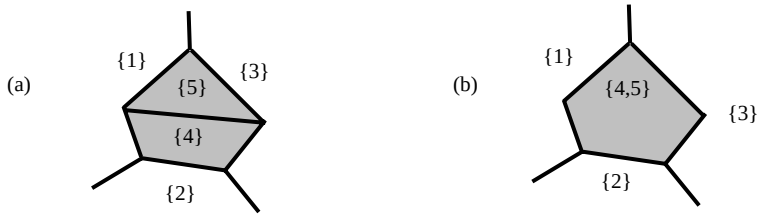
Figure 6-1: Computing neighborhood difference

Whenever two corresponding face-polygons have a neighborhood difference that is not zero, it indicates a conflict.

6.1.2 Merging Regions

In general, merging of two adjacent *rt*-regions is where two distinct *rt*-regions before the merging become indistinguishable after merging. The neighbor ring identifier list of the resultant *rt*-region is changed along with the change in neighbor ring. An example is shown in Figure 6-2. When we merge two constraints $\langle Id_1, B_1, B'_1\rangle \in \mathcal{C}$ and $\langle Id_2, B_2, B'_2\rangle \in \mathcal{C}$, they are replaced with $\langle Id_1 \cup Id_2, B_1 \cup B_2, B'_1 \cup B'_2\rangle$.

Merging of unmatched regions has a special meaning, in that it signifies the relaxation of the constraints around the unmatched regions. Initially, two regions are distinct as a result of the boundaries between the constraints. When merged, the newly formed region is assumed to be included in all the constraints of the regions that were merged.



The above figure shows the changes on the neighbor ring identifier list on one surface by merging. It is similar for the corresponding surface, which is not shown in this figure. All the regions are labeled by their identifiers. In figure (a) on the left, the shaded rt-regions with identifiers {4} and {5} are to be merged. Region with *Id* = {5} and *Id* = {4} have neighbor ring identifier lists ({3},{1},{4}) and ({2},{3},{5},{1}) respectively. Figure (b) on the right shows the result after merging the two shaded rt-regions in figure (a). The merged rt-region have Id={4,5} and a neighbor ring is ({3},{1},{2}).

Figure 6-2: Merging and neighbor ring identifier list

6.2 Inferring Correspondence for Unmatched Regions

In the actual process, correspondences in $\mathcal{C}$ containing unmatched regions that are adjacent are merged. These unmatched regions represent areas of conflicts, and merging them serves to simplify the situation as constraints which involved them will definitely have to be relaxed to make a continuous map possible. For all cases of $\langle Id_i, B_i, \varnothing \rangle, \langle Id_j, B_j, \varnothing \rangle \in \mathcal{C}$, where $i \neq j$ and B_i and B_j are adjacent, merge the two tuples by removing them from $\mathcal{C}$ and replacing them with $\langle Id_i \cup Id_j, B_i \cup B_j, \varnothing \rangle$. Similarly for all cases of $\langle Id_i, \varnothing, B'_i \rangle, \langle Id_j, \varnothing, B'_j \rangle$.

After all adjacent unmatched regions are merged, remove the identifiers of the temporary constraints, i.e. for every $\langle Id_i, B_i, B'_i \rangle \in \mathcal{C}$, replace it with $\{\langle Id_i - Tid, B_i, B'_i \rangle\}$. These identifiers of temporary constraints are no longer needed. This is because the previously unspecified regions are now merged with their neighbors, thus inheriting all their identifier sets.

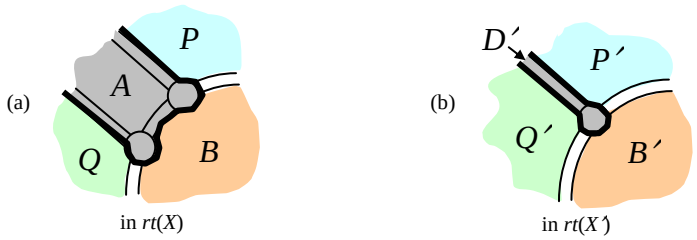
In the worst case, all the edge-quads and vertex-polygons can be merged such that all the regions specified in $\mathcal{S}$ are separated from each other.

In addition, for any $\langle Id_i, B, B' \rangle \in \mathcal{C}$, such that there are more than one component in B and B', it is broken down into $\{ \langle Id_i, B_i, B'_i \rangle : i = 1, 2, \ldots \}$ such that each B_i and B'_i is either one component, or a maximally connected subset, of B and B' respectively, or $\varnothing$, such that $\sum_i Ndiff(B_i, B'_i)$ is minimized. The current implementation is an exhaustive search of all possible pairs of components of B and B' to find the pair of components which is are most similar in their neighborhood and specify a correspondence over them.

After this, it becomes possible to infer correspondence for unmatched regions based on the assumption that the neighbor rings of a pair of corresponding regions should correspond to each other too. For any pair of corresponding regions B and B', if some of their neighbors are unmatched, the correspondence of these unmatched neighbors can be derived. The unmatched regions in the neighbor rings of B and B' can always be matched in the same relative positions exactly: let A, P and Q be regions in $N(B)$

such that $\langle Id_A, A, \varnothing\rangle$, $\langle Id_P, P, P'\rangle$, $\langle Id_Q, Q, Q'\rangle \in \mathcal{C}$, and P, A, Q is consecutive in $N(B)$. Since there is an unmatched region between P and Q in $N(B)$, there must exist some $\langle Id_{A'}, \varnothing, D'\rangle \in \mathcal{C}$ between P' and Q' in $N(B')$. An example is illustrated in Figure 6-3. Even when there is no regions between $rt^{-1}(P')$ and $rt^{-1}(Q')$ on the original surface X', the edge between them becomes a region D' in $rt(X')$ and can thus be corresponded with the region A in $rt(X)$.

An implication of the above is that for two corresponding region B and B' with corresponding neighbor ring identifier lists $Nrid(B)$ and $Nrid(B')$, the numbers of unmatched regions in $N(B)$ and $N(B')$ are the same and occur in the same relative positions to the other neighbors which are corresponding.



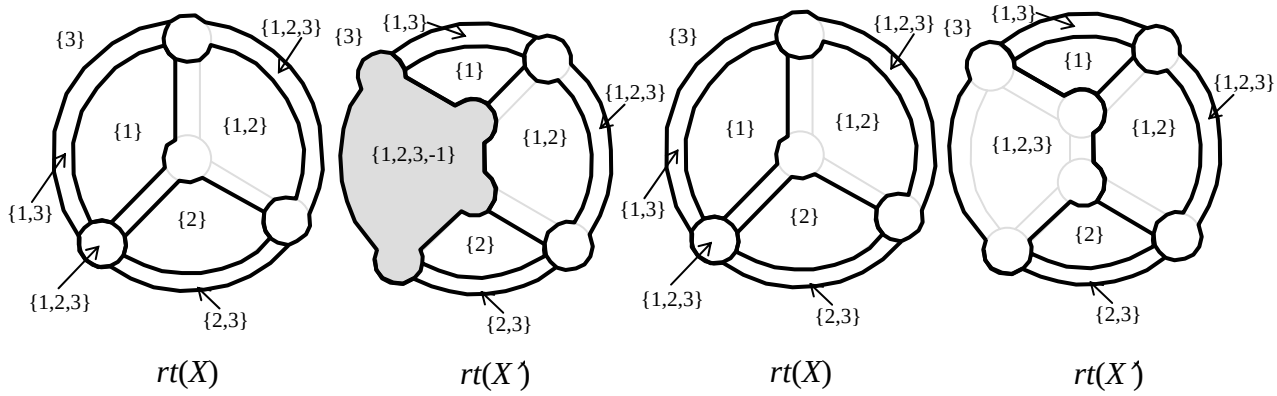
B and B' are corresponding regions under consideration. Given $\langle Id_A, A, \varnothing\rangle$, $\langle Id_P, P, P'\rangle$, $\langle Id_Q, Q, Q'\rangle \in \mathcal{C}$. The gray rt-regions shown above are the resulting unmatched regions from the constraints and are merged into a bigger unmatched region. In figure (a) on the left, P, A, Q is consecutive in $N(B)$, where A now merged with its adjacent edge-quads and vertex-polygons becoming a single unmatched region. In figure (b) on the right, the edge-quad between P' and Q' is also unmatched, as P and Q do not have a common edge-quad in figure (a). Thus, in the neighbor rings of B and B', the unmatched regions A and D', which are in the same relative position about B and B', are derived to be corresponding. Even though $rt^{-1}(D')$ is just an edge in X', working on $rt(X')$ enable it to represent a patch, allowing it to form a correspondence with the region A in $rt(X)$.

Figure 6-3: Neighbor rings of corresponding rt-regions

6.3 Example

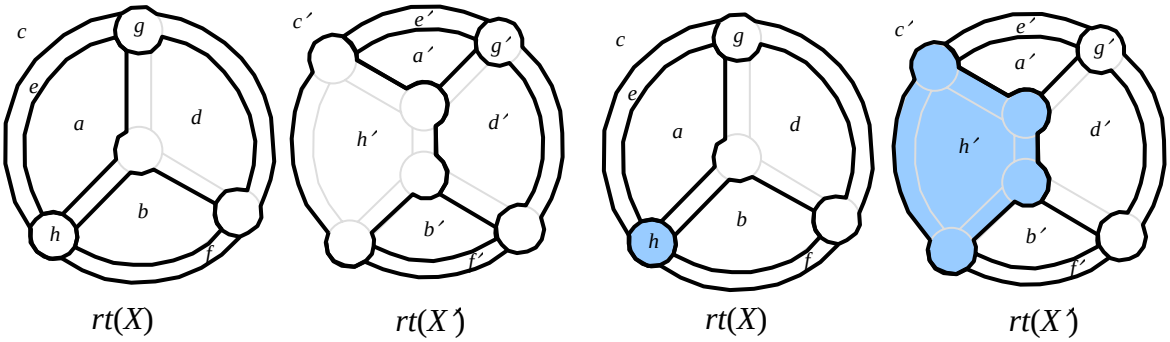
This example continues from Figure 5-2. Figure 6-4(a) shows the result after merging the unmatched regions in Figure 5-2(g). Figure 6-4(b) shows the result after the identifiers for the temporary constraints are removed. Figure 6-4(c) labels all the *rt*-regions for easier reference and also displays the content of complete correspondence set $\mathcal{C}$. Figure 6-4(d) and (e) shows the pairing of the regions and that no further relaxation is required. Figure 6-4(f) shows the modified constraints on the original surfaces represented by the complete correspondence $\mathcal{C}$.

Figure 6-4: Example: Conflict Analysis



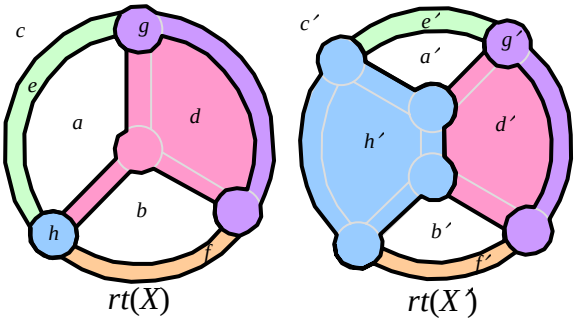
(a) Adjacent unmatched regions (shown in gray in Figure 5-2(g)) are merged to simplify the problem. The resultant unmatched region is shown above in gray.

(b) The temporary identifier -1 is removed.

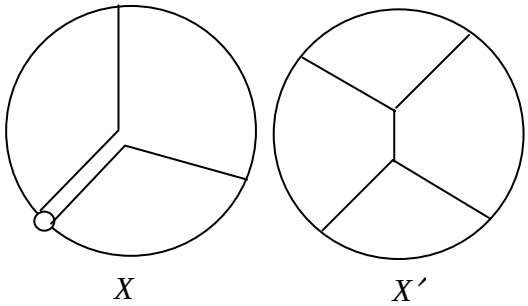


(c) To make illustrations easier, the resulting parts are named as above. After pre-processing, $\mathcal{C}$ = { ⟨{1},*a*,*a*´⟩, ⟨{2},*b*,*b*´⟩, ⟨{3},*c*,*c*´⟩, ⟨{1,2},*d*,*d*´⟩, ⟨{1,3},*e*,*e*´⟩, ⟨{2,3},*f*,*f*´⟩, ⟨{1,2,3},*g*,*g*´⟩, ⟨{1,2,3},*h*,∅⟩ , ⟨{1,2,3},∅,*h*´⟩ }

(d) The correspondence of *h* and *h*´ is inferred from the correspondence of their neighbors.



(e) Since there are no pairs of corresponding face-polygons with neighborhood difference greater than 0, no further relaxation is required. Otherwise, relaxation is done as in Section 7.



(f) This figure shows the required changes to the original constraints/mesh surfaces from the originals shown in Figure 5-2. The visible effect will be the growth of the small newly-created region in the lower-left corner of *X* as we morph *X* to *X*´.

7 CONSTRAINT RELAXATION

This section describes the process of resolving the conflicts that are identified in the previous sections. Section 7.1 gives an overview of the procedure and Section 7.2 describes the actual relaxation process.

7.1 Overview

This stage concentrates on solving the constraints of the form $\langle Id_i, B_i, B'_i \rangle \in \mathcal{C}$, where B_i and B'_i are face-polygons of $rt(X)$ and $rt(X')$ respectively, and $Ndiff(B_i, B'_i) > 0$. These are constraints whose *rt*-regions are corresponding, but have neighbors which do not correspond.

To make $Ndiff(B_i, B'_i) = 0$, the neighborhoods are aligned as much as possible as in the computation of the neighborhood difference (Section 6.1.1). Based on the correspondence of the neighbors, the areas in the neighbor rings which cannot be aligned are identified and the constraints in those areas are relaxed by removing the influences of some specified constraints through merging of the affected regions.

The basic operation in constraint relaxation is a special case of merging regions (Section 6.1.2). Two vertex-polygons incident to a common edge-quad are merged by *collapsing* the edge-quad, forming a new *rt*-region which includes both vertex-polygons and the edge-quad. This new *rt*-region is also regarded as a vertex-polygon. When such merging is done in $rt(X)$, any existing *rt*-regions in $rt(X')$ corresponding to the regions being merged are similarly merged at the same time. This relaxes the constraints influencing the vertex-polygons and edge-quad under consideration without affecting the correspondence of the face-polygons themselves on both surfaces.

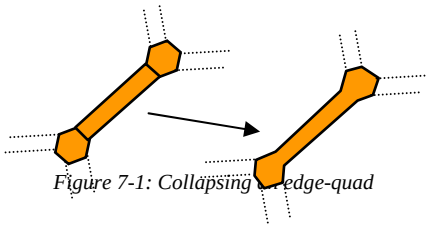


Figure 7-1: Collapsing an edge-quad

Figure 7-1 shows the collapsing of an edge-quad where an edge-quad and its adjacent vertex-polygons are merged into a new vertex-polygon.

7.2 Relaxing Constraints

All pairs of corresponding face-polygons B, B' can be classified according to the neighborhood difference. When $Ndiff(B, B') = 0$, the neighborhood is already locally continuous under the existing correspondence. When $Ndiff(B, B') > 0$, the neighborhoods of B and B' are arranged in some layout used in computing the neighborhood difference that gives the minimum number of unaligned neighbors. All unaligned neighbors are marked as unmatched.

Then, any two vertex-polygons occurring consecutively in the neighborhoods of B, B' which are marked unmatched are merged by collapsing the edge-quad between them. When no more such cases are found, the unmatched vertex-polygons in B will then be corresponded to those in B' according to their relative position with respect to the aligned regions, resulting in two new neighborhoods which now gives $Ndiff(B, B') = 0$. A detailed example is shown in Figure 7-2.

The neighborhood difference function itself can be used as a simple value function to prioritize the selection of corresponding face-polygons for processing. A high neighborhood difference indicates a more complex case where a high number of vertex-polygon merges has to be carried out. As the vertex-polygons being merged also lie in the neighborhood of some other face-polygons, solving a more complex case may indirectly solve other related cases, potentially making the process more efficient. Arbitrary choices may occur when choosing between the various ways of aligning the neighborhoods of a pair of corresponding face-polygons, all of which gives the same number of unaligned elements, as shown in Figure 6-1. This will simply give rise to different results, all of which are acceptable.

The most direct approach is to simply process each case where $Ndiff(B, B') > 0$ such that it becomes 0, through merging of the regions in the neighbor rings. There are basically two levels of consideration:

1. The order in which pairs of corresponding face-polygons B and B' are processed.

2. The manner in which the two neighbor rings are aligned.

Although there is no guarantee of optimal solution, some simple rules of thumb can be used to good results:

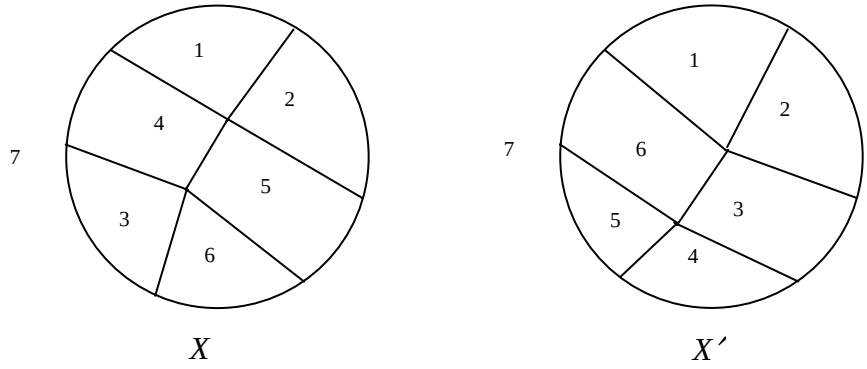
1. When aligning neighbor rings, priority is given to align as many edge-quads as possible (instead of vertex-polygons).

2. Whenever a region has only two neighbors left, the two neighbors are recorded to disallow their merging with each other. When considering the order in which the pairs of corresponding face-polygons are to be aligned, priority is given to those face-polygons with the largest number of neighbors which were involved in such records as these are the cases with the most restrictions to merging.

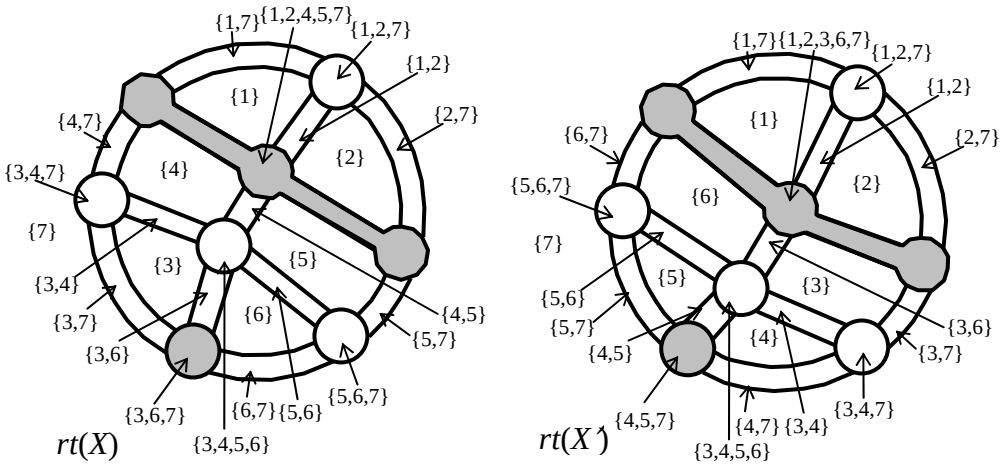
The rationale behind the first rule is to try to limit the number of merges of edge-quads and vertex-polygons to a minimal. This is the same as trying to maximize the number of aligned edge-quads for all pairs of corresponding face-polygons.

For the second rule, some background is necessary. In cases where merging regions with holes (Example in Section 8) are necessary, they will already appear in the result at the end of the Conflict Analysis stage (Section 6). This is because a merged region with holes only arises when some $F \subseteq X$ on one of the original surfaces has absolutely no continuity with respect to $\varphi(F) \subseteq X'$ in the corresponding surface. If such a case exists, the region would have been marked out as unmatched and emerged by the end of the Conflict Analysis stage. All those remaining pairs of regions represented by the pairs of corresponding face-polygons will have some continuous boundary with some other regions. This gives rise to the understanding: for any pair of corresponding face-polygons $B \subseteq rt(X)$ and $B' \subseteq rt(X')$ with exactly two neighbors in each of the corresponding neighbor rings, constraint relaxation will never need to merge the neighbors. This is because merging the two remaining neighbors will result in a merged region with a hole, which is not necessary since it did not appear by the end of Conflict Analysis stage.

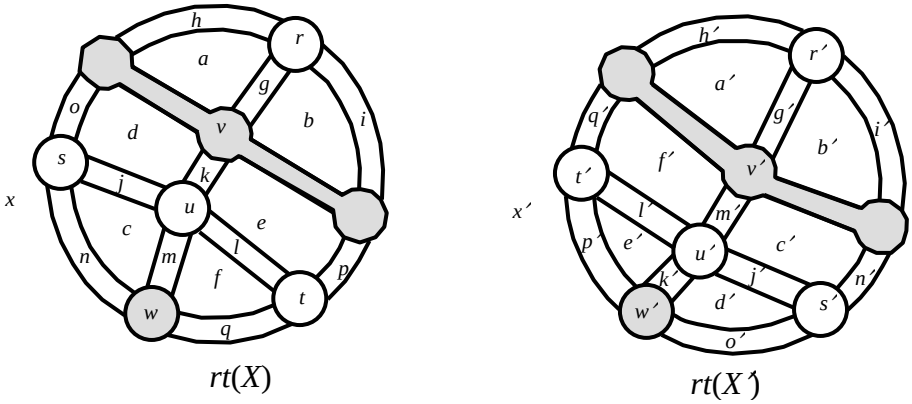
Figure 7-2: Example with Constraint Relaxation



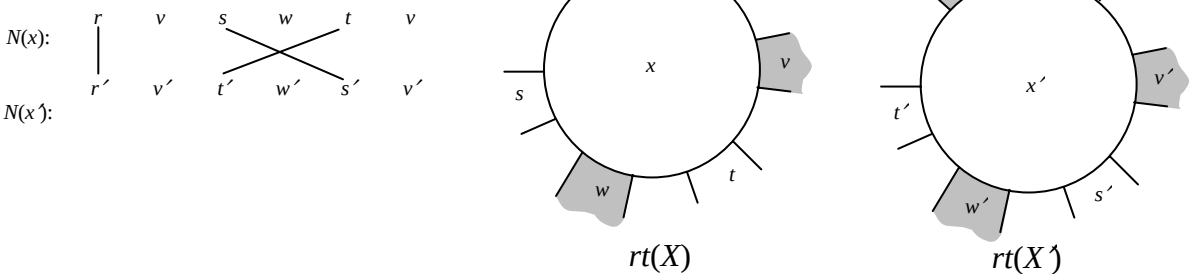
(a) This figure shows the structure of the mesh used to represent two surfaces *X* and *X´*, as well as the specified constraints indicated by the integers 1 to 7.



(b) This figure shows the resultant complete correspondence in *rt*(*X*) and *rt*(*X´*), after the adjacent unmatched regions are merged. The unmatched regions are shown in grey.

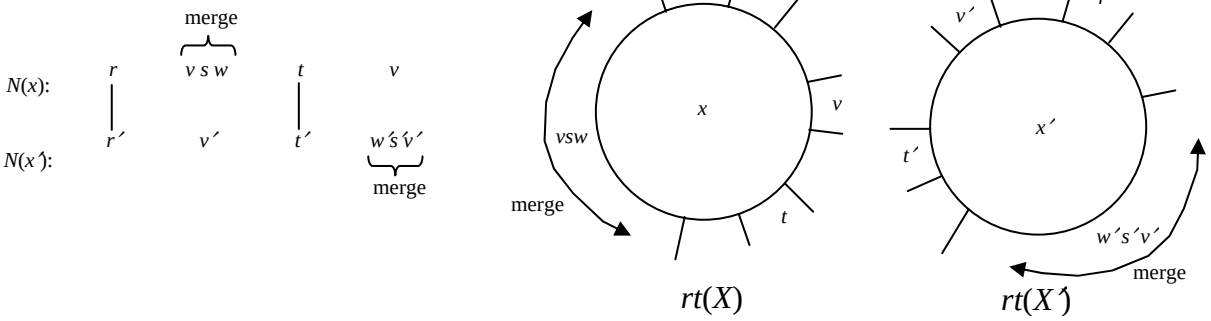


(c) This figure labels the rt-regions for subsequent figures.



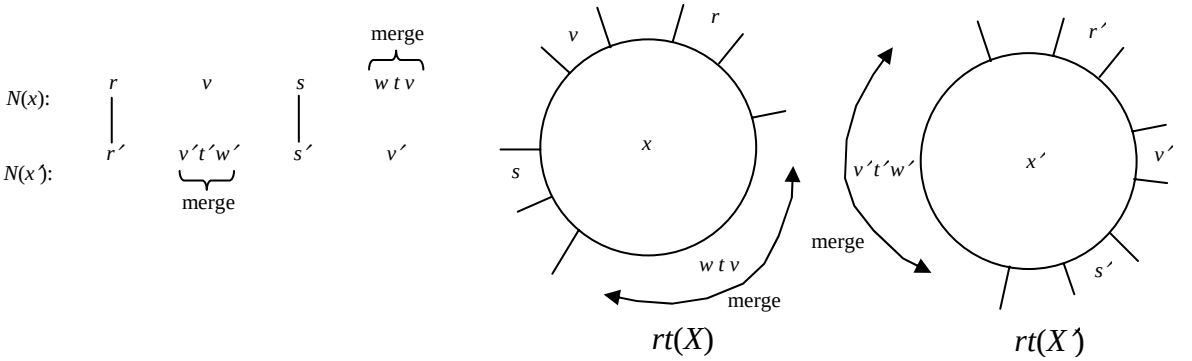
(i) Finding neighborhood difference (ii) Neighborhoods of x and x'.

(d) Taking the counterclockwise direction around regions x and x', (i) shows the neighborhoods, where the lines are drawn to connect vertex-polygons which can be aligned. (ii) shows the regions x and x' and their neighborhoods.



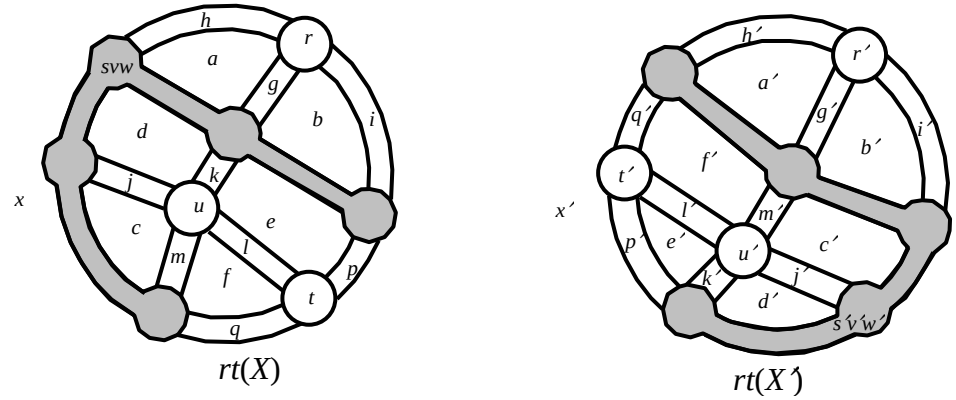
(i) Finding neighborhood difference (ii) Neighborhoods of x and x'.

(e) This is one of the two ways to align the neighborhoods shown in (d)(i) and gives $Ndiff(x,x') = 8$. (ii) shows the corresponding changes to the neighborhoods of x and x' after the unaligned vertex-polygons are merged.

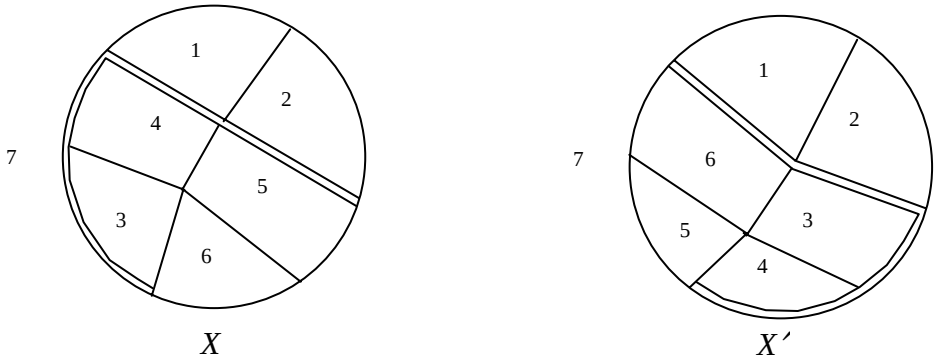


(i) Finding neighborhood difference (ii) Neighborhoods of x and x'.

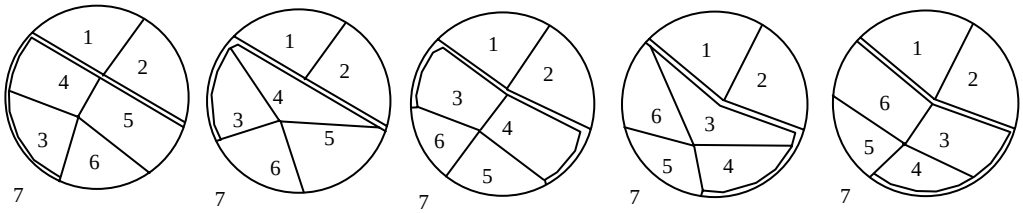
(f) This is the other alternate way to align the neighborhoods shown in (d)(i) which also gives $Ndiff(x,x') = 8$.. (ii) shows the corresponding changes to the neighborhoods of x and x' after the unaligned vertex-polygons are merged.



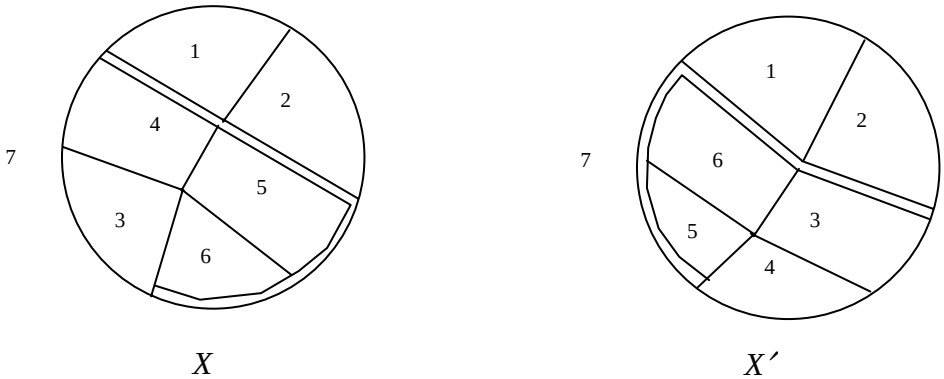
(g) This figure shows the relaxed regions in grey, which result from the merging done in (e).



(h) This figure shows the changes to be original constraints/mesh surfaces required. One way to show continuity will be to morph *X* to *X´* according to the revised constraints, which can be shown as a clockwise swirling of the combined region indicted by 3,4,5,6. The added buffer region will resolve the conflict this swirling region have with the regions indicated by 1 and 2. The next figure shows the transition.



(i) This figure shows how the regions labeled 4,5,6 and 7 will be swirled in clockwise direction while maintaining continuity if we morph *X* to *X´*, after the constraints are relaxed as indicated in figure (h) above.



(j) This shows the alternative solution, which is derived from merging done in (f), by collapsing edges *p*, *p´*, *q* and *q´*. This solution will result in a counter-clockwise swirling of the regions labeled as 4,5,6 and 7 as opposed to the clockwise swirling shown in (i).

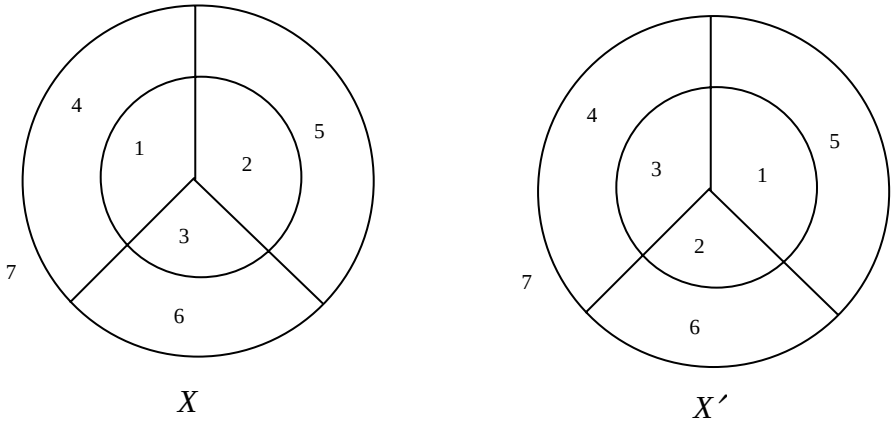
8 FURTHER EXAMPLES

This section shows two more examples which are compound versions of the basic examples shown in the previous sections.

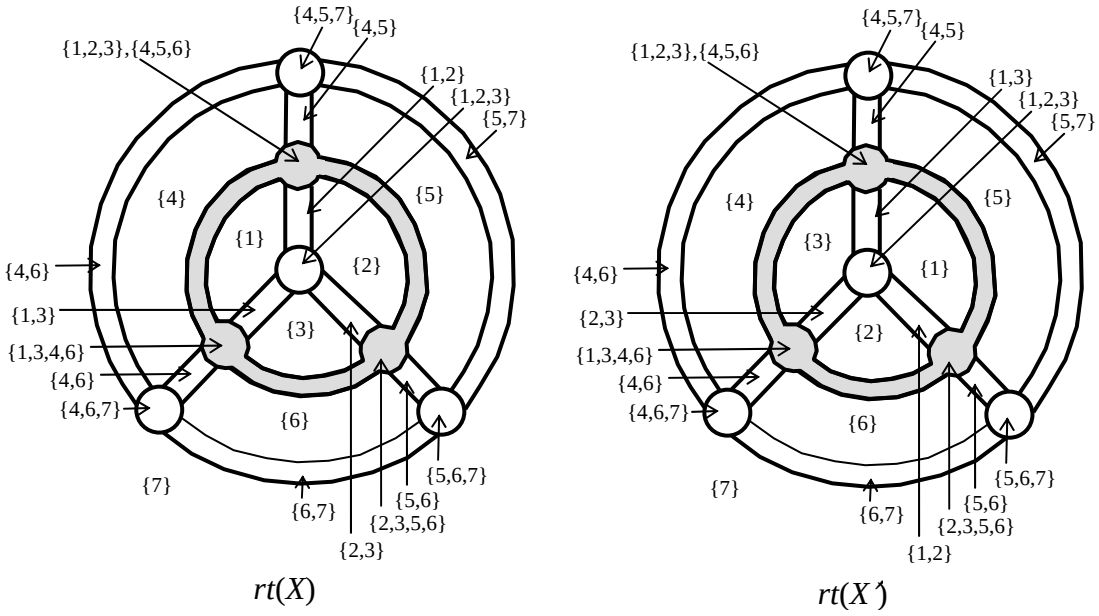
8.1 Division into Sub-Problem

There is a special case when the constraints specified over the two surfaces are so different that after the constraints processing phase (Section 5) some merged regions form loops on the surfaces. This indicates that the original problem can be split into simpler sub-problems and solved as such before combining the individual solution into the final solution. This is shown in Figure 8-1 as follows.

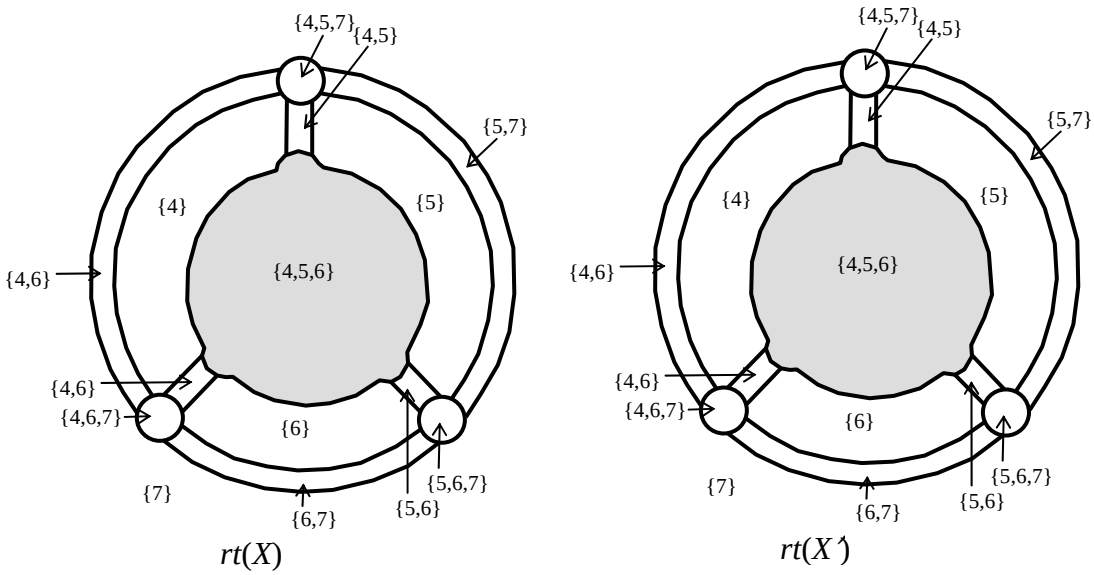
Figure 8-1: Example with division into sub-problems



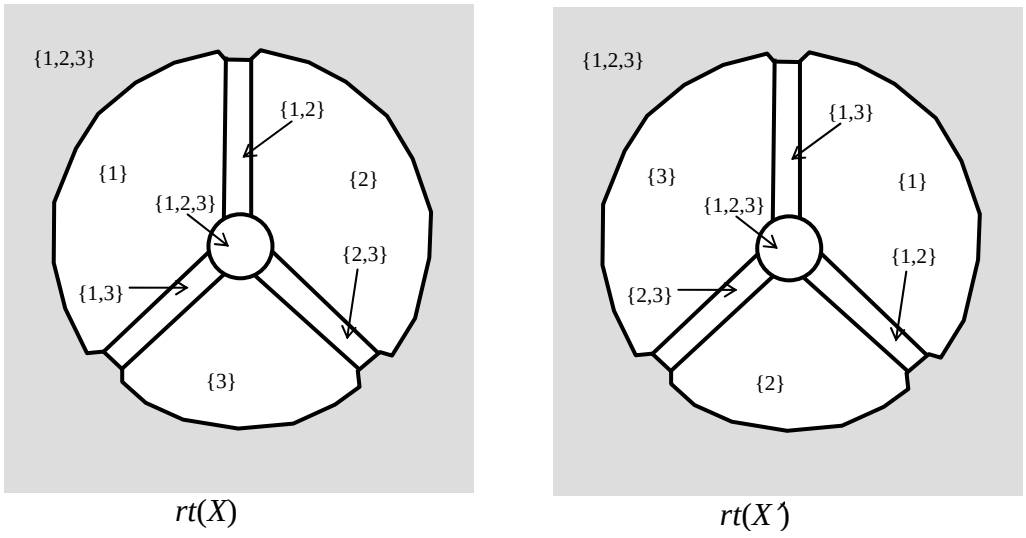
(a) This figure shows the structure of the mesh used to represent two surfaces X and X', as well as the specified constraints indicated by the integers 1 to 7.



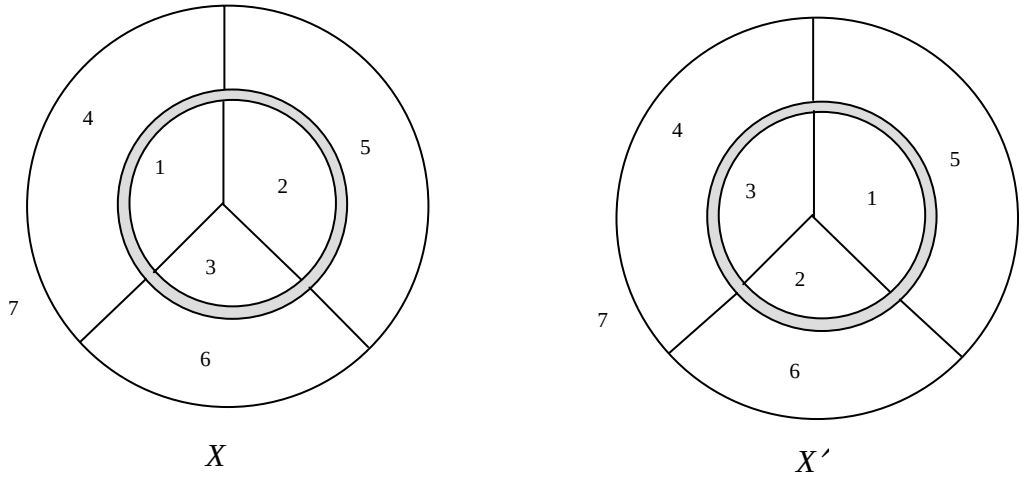
(b) This figure shows the resultant complete correspondence in $rt(X)$ and $rt(X')$, after the adjacent unmatched regions are merged. The unmatched regions are shown in grey. In this case, the unmatched region on each surface is a ring and has two boundaries, which effectively sub-divide the problem into two portions: one with the unmatched regions and the regions 1,2 and 3 and another with the unmatched regions and the regions 4,5,6 and 7. The two sub-problems are resolved independently as shown in the following figures: (c) shows sub-problem 1; (d) shows sub-problem 2.



(c) Sub-problem 1: The *rt*-regions inside the unmatched ring are ignored. With respect to this sub-problem, the neighborhood of the unmatched region in the center is (4,6,5).



(d) Sub-problem 2: The *rt*-regions outside the unmatched ring are ignored. With respect to this sub-problem, the neighborhood of the unmatched region is (1,2,3).



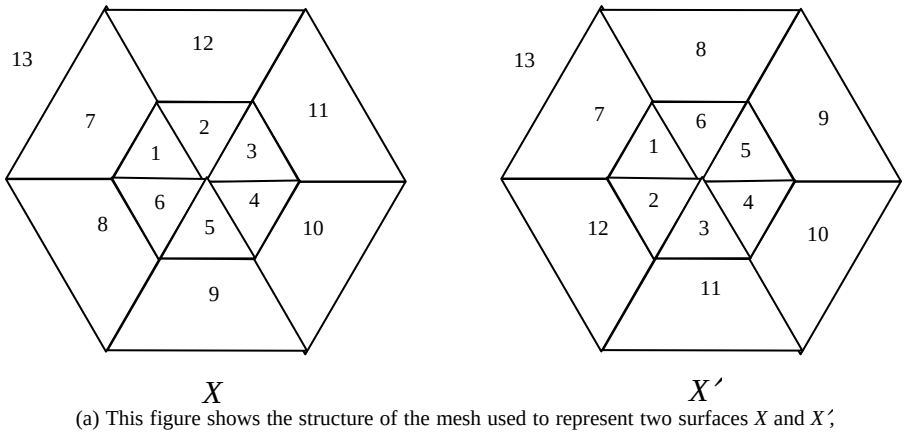
X *X´*

(e) This figure shows the changes to the original constraints/mesh surfaces required. A ring buffer is introduced that separates the each surface into two non-interacting parts. The overall effect if we were to morph *X* to *X´* according to the revised constraints will be a swirling of center part consisting of regions 1,2 and 3, either clockwise or counter-clockwise, while the regions 4,5,6 and 7 remains the same, and the grey regions acting as buffers.

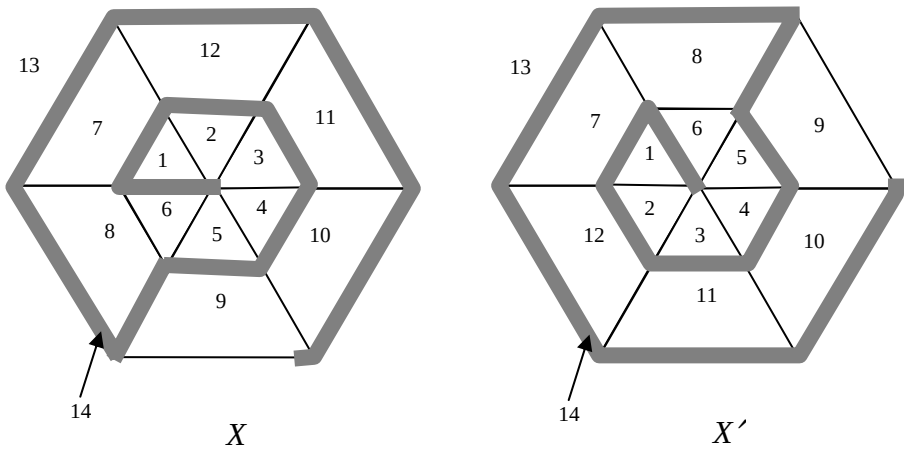
8.2 A More Complex Example

There is a more complex example which illustrates that the simple rules of merging neighbor rings in order to preserve continuous boundaries between corresponding regions. This is shown in Figure 8-2 as follows. In this example, although the simple connectivity of individual regions in *X* and *X´* are similar, the ordering (counter-clockwise) is different. Thus, buffers has to be introduced in order to achieve a common topology in the constraints.

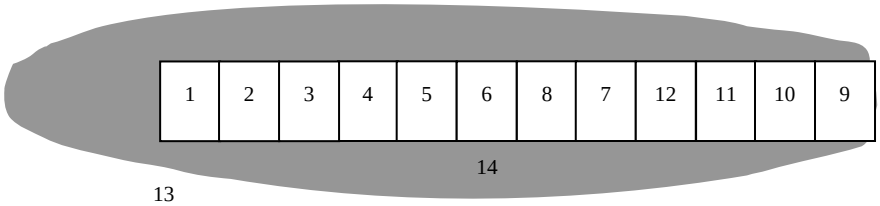
Figure 8-2: A more complex example



(a) This figure shows the structure of the mesh used to represent two surfaces *X* and *X´*, as well as the specified constraints indicated by the integers 1 to 13.



(b) This figure shows one possible solution, where the changes to the original constraints/mesh surfaces required are shown in grey. The grey region is introduced to act as a buffer in order to resolve the conflicts to continuity that arises from the original constraints.



(c) The common topology of the *X* and *X´* after the constraints are changed in (b).

9 GENERALIZATION

The basic solution mentioned in the previous sections assumed that the original constraints are specified over topological disks. In this section, we examine the problems and solutions on the cases where the regions can be of other topologies and how the problem can be preprocessed so that the basic solution can be applied.

9.1 Degenerate Cases

The *genus g* of a region *A* is the number of openings in a sphere, to which the closure of the *A* is topologically equivalent. It is defined by the following form of Euler's Characteristic:

$$|vertices(cl(A))| - |edges(cl(A))| + |faces(cl(A))| = 2 - g$$

g	***A*'s topology**
0	sphere
1	disk
2	open cylinder
…	…

High genus in this context refers to cases where $g > 1$. A *hole* in this context refers to such an opening.

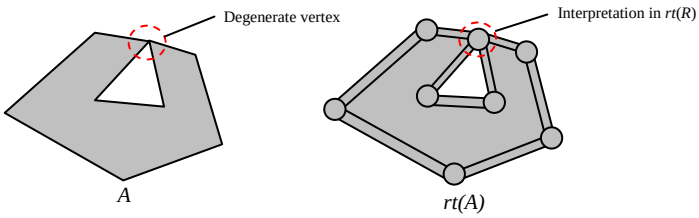


Figure 9-1: Degenerate cases in regions

If constraints specified on the original surface are not proper disks, there may be degenerate cases as shown in region *A* above. This can be avoided if *A* is interpreted as the corresponding *rt*(*A*) instead, which avoids degenerate cases by treating vertices and edges as faces.

9.2 Constraints on High-Genus Regions

Allowing high-genus regions to be specified in the constraints introduces a number of problems which cannot be solved directly by the basic method. High-genus regions introduces multiple holes, each of which is the boundary between *rt*(*A*) and some subset of *rt*(*X*) – *rt*(*A*) , which is topologically a disk. For a continuous map to be possible between two *rt*-regions $P \subseteq rt(X)$ and $Q \subseteq rt(X')$, and let the *rt*-regions bounded by the *n* holes of *P* be $\{P_1, P_2, \ldots P_n\}$ and the *rt*-regions bounded by the *m* holes of *Q* be $\{Q_1, Q_2, \ldots Q_m\}$ the following condition must be satisfied:

C1. $n = m$ and there is a bijective mapping $\{(P_i,Q_i) : i = 1, 2, \ldots, n\}$ such that there is a continuous map between P_i and Q_i under complete constraint set $\mathcal{C}$. A necessary condition is that for all $\langle Id, B, B' \rangle \in \mathcal{C}$, if $B \subseteq P_i$, then $B' \subseteq Q_i$.

9.3 Conflict Between Hole-Bounded Regions

In Figure 9-2, *P* and *Q* are corresponding. Suppose P_i and P_j are corresponded to Q_i and Q_j respectively under some constraints, and lie in some hole-bounded *rt*-regions shown above, a continuous map is not possible because condition C1 is not met. In such a case, relaxation of the boundary of constraints is insufficient to get a solution. As such, modification of the constraint set is necessary. There are basically two choices as follows:

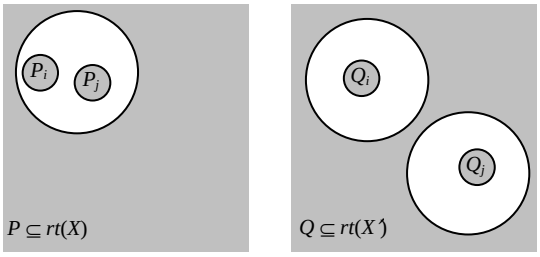


Figure 9-2. Conflict between hole-bounded regions

Remove the Constraints that Result in such Conflicts

The constraints within the complete constraint set that result in such cases are to be removed. However, there is problem of arbitrary choices in the removal. As in Figure 9-2, only one of the two constraints (P_i, Q_i) and (P_j, Q_j) needs to be removed. To ensure minimal changes to the constraint set, ideally, a global minimum solution should be found that removes the least number of constraints such that condition C1 is satisfied. The constraints are removed by merging regions. The merging process will eventually terminate, as the worst case is when there is only one constraint left.

Merging Holes

In the case shown in Figure 9-2, a solution can be found by *hole-merging*, which merges the holes in P and Q until no more such conflicts are found. For this example, P_i and P_j are corresponding to Q_i and Q_j respectively. Since P_i and P_j lie in the same hole-bounded region in P, but Q_i and Q_j lie in different hole-bounded regions in Q, we can resolve this by merging the two holes in which Q_i and Q_j lie in. This is done by introducing a cut in Q that joins the two holes, as shown in Figure 9-3. The path of cutting should be chosen in such a way that the neighborhood of the hole is preserved as much as possible.

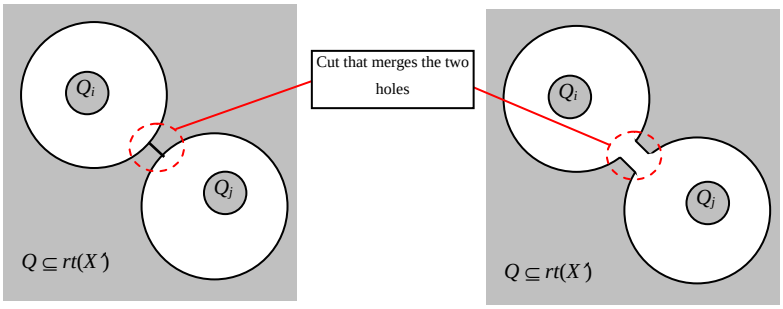


Figure 9-3: Merging holes

The merging of holes will eventually terminate, as in the worst case, all holes are merged into one, and thus P and Q topologically disks.

9.3.1 Unmatched Holes

After the conflicts between hole-bounded regions are resolved, the only remaining possibility of contradicting C1 is when some of the hole-bounded regions are unmatched under the constraints. To make a continuous map possible, the constraint set is modified by creating holes in the corresponding region.

9.3.2 Application of the Basic Solution

After C1 is met and the mapping of the multiple neighbor rings is known, the corresponding neighbor rings are resolved in a similar method as mentioned in the basic method. Essentially, when considering the correspondence of a hole and its bounded region, the other holes and hole-bounded regions can be temporary disregarded with respect to the current context, effectively making it topologically a disk, and the basic solution can then be applied. Similarly, the other holes can be solved individually as independent problems, before combining the solutions into a continuous map over the original surfaces. This is similar to division into sub-

problems described in Section 8, except that the division here is chosen as a way to simplifying the problem and not a direct result of constraints as in Section 8.

10 LIMITATION AND FUTURE WORK

This work relies on several assumptions to guarantee that a solution can be found. In this section, these assumptions, which are also the limitations, are discussed along with the implication of lifting them.

10.1 Constraints on Vertices or Edges

In this work, it is assumed that constraints are only specified over faces in the original mesh. By re-tessellating the original mesh and allowing vertices and edges to be treated as vertex-polygons and edge-quads, continuity problems that result from constraints specified over faces can be avoided by changing the influence of constraints over the vertex-polygons and edge-quads. However, if constraints can be specified over vertices and edges on the original surfaces, a continuous map cannot be guaranteed by the same method.

However, the method can be adapted such that each specified constraint over vertices and edges can be treated as constraints over very small faces and very narrow faces respectively. This interpretation will then make the new problem similar to the basic problem described within this thesis. This can be done by re-tessellating the original mesh twice. A mesh X is re-tessellated into $rt(cl(rt(X)))$. $rt(X)$ gives the sets of faces corresponding to all simplices in X. $cl(rt(X))$ closes this set of faces so that it becomes a proper water-tight mesh, which is then treated as the secondary input mesh, to be re-tessellated again. Similarly the original constraints also go through two layers of translation. A constraint (Y,Y') for $Y \in X$ and $Y' \in X'$ becomes $(rt(cl(rt(Y))), rt(cl(rt(Y'))))$ in the twice-tessellated surfaces of X and X'.

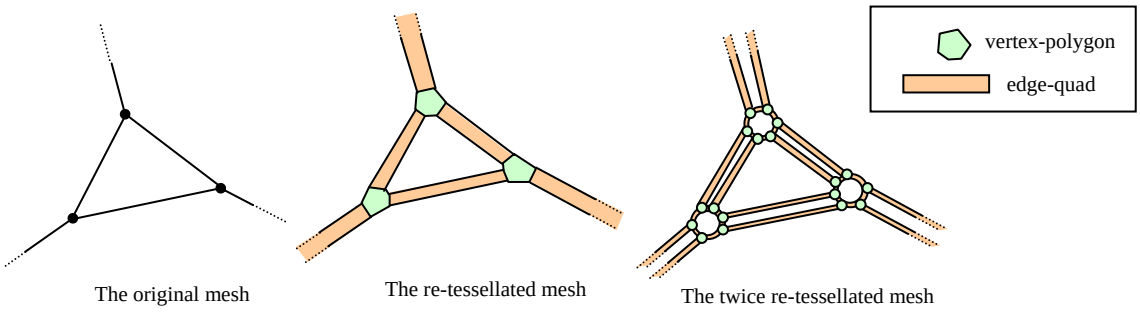


Figure 10-1: Twice re-tessellated surface

10.2 Non-watertight Surfaces

Another assumption is that the two surfaces to be corresponded must be watertight. If the two surfaces are topologically equivalent to spheres with holes, the holes can be treated as faces in the procedure, but must be distinguished from ordinary faces in that faces representing holes can only be corresponded with faces representing holes. Thus, an adaptation of the basic solution is required.

10.3 Topology Changes

When two surfaces are of different topology (such as different genera), there is no continuous map. Thus, the work described herein is not directly applicable, since it relies on the assumption that the two surfaces are of the same topology.

However, if the two original surfaces can be preprocessed to become homeomorphic, either automatically or specified explicitly, it can then be treated as finding a continuous map between two non-manifold surfaces. In addition, the complete correspondence set computed by this work can be helpful in discovering how to automatically detect and best handle topology changes of two surfaces under given constraints. The details awaits further work.

10.4 Extension to Higher Dimensions

The solution discussed so far deals with two closed surfaces. Extension of the basic idea into higher dimension, such as closed volumes, looks promising by adopting a recursive view to the problem. The high dimension problem is recursively broken down into lower dimension problems, and the solutions from lower dimension problems can be combined into a solution for high dimension. This makes a very interesting avenue awaiting further exploration. The work presented in this thesis is a 2-D problem in the following classification.

1-D problem : Given two closed lines (rings) and a set of constraints that correspond segments of the two lines, find a continuous map from one line to the other line. Each line can be divided into discrete segments according to the constraints. A continuous map over the whole line can be achieved by making sure that each pair of corresponding line segments is continuous at the two ends of each segment.

2-D problem : Given two closed surfaces, find a continuous map from one surface to

the other given some correspondences over regions, which are assumed to be simple. A continuous map over the whole surface is achieved by making sure that each pair of corresponding patches has a continuous map around their boundaries, which is the 1-D problem mentioned earlier.

3-D problem : Given two closed volumes, find a continuous map from one volume to the other given correspondences over simple polytopes. For two simple polytopes that are specified to be corresponding, there must be a continuous map on its surface, which is actually the 2-D problem.

Although high dimension solutions seem feasible theoretically, adaptations have to be made such that changes or decisions at a particular lower-dimension are propagated back to the higher dimension. For example, when considering 3-D problems and computing a continuous map for the surface of two corresponding polytopes, any merging of regions on the surface has to be propagated to the polytopes whose faces are regions in the 2-D problem.

11 CONCLUSIONS

Given a set of constraints over two genus 0 manifolds, it is not always possible to achieve continuous map. This thesis presents a basic solution that automatically identifies where the conflicts occur and relaxes the associated constraints such that a continuous map can always be achieved.

Although there is no proof of optimality in our minimization of relaxation, our implementation results show a general agreement between the computed solutions and solutions constructed by a human expert.

Application-wise, 3D surface morphing is the major field for this work. It can be introduced as an intermediate module between high-level human or automatic algorithms and low-level morphing techniques as in [ZOT03]. Given some high-level inputs (constraints), it can resolve all the conflicts to getting a continuous map, and in the process, simplify the problem. As this work presents a fundamental idea in achieving continuity in morphing with respect to user wishes, rather than an individual morphing technique, the approach can easily be incorporated with standard mapping techniques into morphing applications.

BIBLIOGRAPHY

[A00] M. Alexa, “Merging Polyhedral Shapes with Scattered Features”, The Visual Computer, v16(1), 26-37, 2000.

[A01] M. Alexa, “Mesh Morphing”, Proceedings of Eurographics, State of the Art Report, 2001.

[A02] M. Alexa, “Recent Advances in Mesh Morphing”, Computer Graphics Forum, v21(2), 173-196., 2002.

[ASS93] B. Aronov, R. Seidel and D. Souvaine, “On Compatible Triangulations of Simple Polygons”, Computational Geometry: Theory and Applications 3, 1, 27-36, 1993.

[BSW97] M. Babikov, D. Souvaine and R. Wenger, “Constructing Piecewise Linear Homeomorphisms of Polygons with Holes”, Proceedings of the 9th Canadian Conference on Computational Geometry, 1997.

[DEG99] T. Dey, H. Edelsbrunner and S. Guha, “Computational Topology”, Advances in Discrete and Computational Geometry (Comtemporary mathematics 223), ed. B. Chazelle, J.E. Goodman and R. Pollack, American Mathematical Society, 109-143, 1999.

[DG96] D. DeCarlo and J. Gallier, “Topological Evolution of Surfaces”, Proceedings of Graphics Interface, 194-203, 1996.

[G86] D. Gillman, “Triangulations for Rubber-Sheeting”, Proceedings of Auto Cartography 7, 191-197, 1985.

[GSL98] A. Gregory, A. State, M.C. Lin, D. Manocha and M.A. Livingston, “Feature-based Surface Decomposition for Correspondence and Morphing between Polyhedra”, Proceedings of Computer Animations, 64-71, 1998.

[GW95] H. Gupta and R. Wenger, “Constructing Piecewise Linear Homeomorphisms of Simple Polygons”, J. Algorithms 22, 142-157, 1995.

[KSK00] T. Kanai, H. Suzuki, F. Kimura, “Metamorphosis of Arbitrary Triangular Meshes”, IEEE Computer Graphics and Applications, v20(2), 62-75, 2000.

[LDS99] A.W.F. Lee, D. Dobkin, W. Sweldens, and P. Schroder, “Multiresolution Mesh Morphing”, Proceedings of ACM SIGGRAPH, 343-350, 1999.

[LV98] F. Lazarus and Verroust, “Three Dimensional Metamorphosis: A Survey”, The Visual Comp0uter, v14(8/9), 373-389, 1998.

[MKF01] T. Michikawa, T. Kanai, M. Fujita and H. Chiyokura, “Multiresolution Interpolation Meshes”, Proceedings of Pacific Graphics, 60-69, 2001.

[PSS01] E. Praun, W. Sweldens and P. Shröder, “Consistent Mesh Parameterizations”, Proceedings of ACM SIGGRAPH, 179-184, 2001.

[SW94] D. Souvaine and R. Wenger, “Constructing Piecewise Linear Homeomorphisms”, Technical Report 94-52, DIMACS, New Brunswick, New Jersey, 1994.

[WG85] M.S. White Jr and P. Griffin, “Piecewise Linear Rubber-sheet Map Transformation”, Amer. Cartographer 12, 123-131, 1985.

[ZOT03] Y. Zhao, H.Y. Ong, T.S. Tan, Y. Xiao, “Interactive Control of Component-based Morphing”, Eurographics/SIGGRAPH Symposium on Computer Animation, 339-348, 385, 2003.

[ZSH00] M. Zöckler, D. Stalling and H. Hege, “Fast and Intuitive Generation of Geometric Shape Transitions”, Visual Computer, v16, 241-253, 2000.