Matability of Polygons

Aya Steiner
Matability of Polygons

Research Thesis

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

Aya Steiner

Submitted to the Senate of
the Technion — Israel Institute of Technology

Adar 5765 Haifa February 2005
The Research Thesis was done under the supervision of Prof. Gill Barequet in the Faculty of Computer Science.

I would like to thank my supervisor Gill Barequet for choosing me to be his student. I thank him for the opportunity to do a high level research, for his assistance and guidance, and for his intellectual and moral support.

The generous financial help of the Technion during my studies is gratefully acknowledged.
Contents

Abstract 1

1 Introduction 3

2 Preliminary Results 9

3 Sufficient Conditions for Matability 15
   3.1 Natural Classes of Matable Polygons 15
      3.1.1 Star-shaped polygons 16
      3.1.2 3-partition polygons 18
   3.2 Pairs of Matable Polygons 22

4 Nonmatable Polygons 31
   4.1 General Description 32
   4.2 Properties 32
   4.3 Nonmatability of $P$ and $Q$ 38
   4.4 A Family of Nonmatable Pairs of Polygons 47
List of Tables

4.1 Specification of $P$ ......................................................... 34

5.1 Performance of the algorithm (empty entries are practically 0) . . . . 75
# List of Figures

2.1 Label sequence ........................................... 10
2.2 A ray shadow ............................................. 11
2.3 Strip and cone shadows ................................. 12
2.4 A double-cone connection .............................. 13
3.1 Starshape polygons always mate ...................... 17
3.2 A 3-partition polygon ................................. 19
3.3 The overlay of any couple of sectors has one of two forms ..... 20
3.4 Two non-intersecting fans having the edge $p_{i_{k+1}}q_{j_k}$ in common .... 21
3.5 A direction along which the points $q_{j_k}$ and $p_{i_{k+1}}$ are the leftmost and rightmost of their respective polygons .......................... 21
3.6 Parallel-separable polygons ............................. 24
3.7 A double-cone between $P$ and $Q$, built upon $p_iq_j$ .............................. 25
3.8 A double-cone between $P$ and $Q$ .......................... 26
3.9 Building the shadow ...................................... 27
3.10 The shadows of $p_i$ and $q_j$ with respect to $\triangle q_jq_{j+1}p_{i-1}$ and $\triangle p_ip_{i-1}q_{j+1}$, respectively ............................................ 28
4.1 The 45-vertex polygon $P$ ......................................................... 33
4.2 The transition vertex between $(q_1, q_2)$ is on a bay (Lemma 4.1) .... 35
4.3 The transition vertex between $(q_1, q_2)$ is on an anvil (Lemma 4.1) . 36
4.4 The transition vertex between $(q_1, q_2)$ is on a spike (Lemma 4.1) . 36
4.5 The $!q_1$ and $!q_2$ shadows are drawn in grey, and their intersection is the
dark grey $q_0$-region (Lemma 4.2) .................................................. 37
4.6 All three anvils are connected to their closest vertices of $Q$ ............ 38
4.7 The $!q_2$ and $!q_0$ shadows and their intersection, all vertices in the dark
grey region are connected to $q_1$ .................................................... 40
4.8 The $!q_2$ shadow cast by the triangle $\triangle p_8 p_9 q_0$ (Lemma 4.3) ...... 41
4.9 The $!q_2$ shadow cast by the triangle $\triangle p_8 p_9 q_1$ (Lemma 4.3) ...... 42
4.10 The $!q_2$ shadow cast by the triangle $\triangle p_8 p_9 q_1$ (Lemma 4.4) ...... 44
4.11 The $!q_0$ shadow covers the vertex $p_{21}$ and enforces one of the vertices
of $A_1$ to be a transition vertex ....................................................... 45
4.12 The $!q_2$ shadow cast by the triangle $\triangle p_8 p_9 q_1$ ......................... 46
4.13 A 63-vertex polygon with a triangle ............................................ 47
4.14 Execution of the algorithm .......................................................... 51
4.15 Family of nonmatable polygons ................................................... 57

5.1 Cleaning the shadow $!p_i$ ......................................................... 61
5.2 Active cells in the overlay of two slices ........................................ 63
5.3 Active cells: Their straight skeletons and triangulations .................. 65
5.4 Trees in the skeleton, rooted at points (shown as black disks) equidistant from the two polygons of the two slices . . . . . . . . . . . . . . 67
5.5 Setting vertex heights according to their offsets . . . . . . . . . . . . 68
5.6 Triangles sharing a contour-intersection point . . . . . . . . . . . . . 69
5.7 A simple interpolation case . . . . . . . . . . . . . . . . . . . . . . . . 72
5.8 A complex branching example . . . . . . . . . . . . . . . . . . . . . . . 73
5.9 A fully reconstructed pair of lungs . . . . . . . . . . . . . . . . . . . . 74
5.10 (a,b,c) Skeleton creation time, number of skeleton edges, and number of output triangles, respectively (in the lungs model), as functions of the number of input contour edges; (d) Number of output triangles as a function of intermediate skeleton edges. . . . . . . . . . . . . . 76
5.11 A reconstructed pelvis . . . . . . . . . . . . . . . . . . . . . . . . . . 77
Abstract

Interpolating a piecewise-linear triangulated surface between two parallel slices, each consisting of an arbitrary number of (possibly nested) polygons that define material and non material regions, has attracted a lot of attention in the literature in the past two decades. This problem has important applications to medical imaging, digitization of objects, and geographical information systems. Data obtained by medical imaging apparata, range sensors, or elevation contours are interpolated in order to represent, reconstruct, and visualize human organs, CAD objects, topographic terrains, etc. Practically all currently-known surface reconstruction from parallel slices methods take the existence of a mating of the polygons for granted. That is, they assume a priori the existence of a non-self-intersecting triangulated surface defined over the vertices of the two polygons, and connecting between them. Gitlin et al. [GOS96] were the first to specify a nonmateable pair of polygons. They verified the nonmateability of those polygons by using computer software. A related interesting question is, given two nonmateable polygons, whether or not one is able to add vertices on the boundary of the polygons so as to make them mateable, and if so, how many such vertices suffice to guarantee a mating. Geiger [Ge93] proved that adding two Steiner points, one on each polygon, allows any pair of polygons to mate.

In this thesis we provide proof of the nonmateability of a simple pair of polygons. The complexity of our pair of polygons is smaller than that of the example of [GOS96].
We also provide, for the first time, a family of nonmatable pairs of polygons, with unbounded complexity. We also give a few sufficient conditions for polygon matability. We show that when the addition of Steiner vertices on the boundary of the polygons is permitted, if one polygon is convex, then one Steiner point suffices to guarantee mating. (If the two polygons are convex, then no Steiner vertices are needed at all.)

We also present an efficient method for interpolating a piecewise-linear surface between two polygons that define ‘material’ and ‘nonmaterial’ regions. Our method is fully automatic and is guaranteed to produce non-self-intersecting surfaces in all cases regardless of the their complexity and geometry. The method is based on computing the cells of the symmetric difference of the two polygons. Then, the straight skeletons of the selected cells guide the triangulation of each face of the above cells. Finally, the resulting triangles are lifted up in space to form an interpolating surface. We provide some experimental results on various complex examples to show the good and robust performance of our algorithm.
Chapter 1

Introduction

The reconstruction of a polyhedral surface from a sequence of parallel polygonal slices has been an intriguing problem during the last thirty years. This problem arises primarily in the fields of medical imaging, digitization of objects, and geographical information systems. Data obtained by medical-imaging apparatus, range sensors, or elevation contours are interpolated in order to represent, reconstruct, and visualize human organs, CAD objects, or topographic terrains. It is assumed that a preprocessing step has already extracted from the raw data (usually a sequence of pixel images) the closed two-dimensional contours, which delimit the material region on each slice. Then the goal becomes to compute a surface that tiles between these contours and forms a solid volume whose cross-sections at the given heights identify with the input slices.

Various algorithms for two-dimensional based polyhedral surface reconstruction have been suggested in the literature (e.g., [Ke75, FUKU77, GD82, WA86, KF88, KSF88, SP88, WW94]). Only a few algorithms [Bo88, BG92, BS96, BCL96, OPC96, CD99, BGLS04] attempt to handle the interpolation problem in full generality. Most algorithms [CS78, CCLB80, SH81, Or81, SP87, ZJH87, MSS92, CP94] reduce the
more involved case to the simple case where each slice contains only one contour. In this thesis we restrict ourselves to the interpolation between two slices, each containing a single polygon. We explore the matability of such a pair by considering several problems related to the existence of a non-self-intersecting triangulated surface defined over the vertices of the two polygons, and connecting between them.

**Mating Existence** Practically all currently-known surface reconstruction from parallel slices methods take the *matability* of the polygons for granted. That is, they assume *a priori* the existence of a non-self-intersecting triangulated surface defined over the vertices of the two polygons, and connecting between them. The primary concern in the literature has usually been to find fast heuristics to select a “good” reconstruction among the many available solutions. A variety of criteria have been used to assess the quality of a particular solution, e.g., the volume of the resulting polyhedral object [Ke75], the total surface area of the added “tiles” [FKU77], the length of the added diagonals [CS78], consistency between the incrementally-constructed perimeter [GD82], and consistency between angle sequences [WW94]. Only in 1996 Gitlin et al. gave the first example of two nonmatable polygons, and verified this example by using computer software. In this thesis we provide proof of the nonmatability of a simpler pair of polygons. The complexity of this problem is smaller than that of the example of [GOS96]. We also provide, for the first time, a family of nonmatable pairs of polygons with unbounded complexity. However, in general, there exist many surfaces interpolating between a pair of polygons. The primary concern in the literature has usually been to find fast heuristics to select a “good” reconstruction among the many available solutions. A variety of criteria have been used to assess the quality of a particular solution, e.g., the volume of the resulting polyhedral object [Ke75], the total surface area of the added “tiles” [FKU77], the length of the added diagonals [CS78], consistency between the incrementally-constructed
perimeter [GD82], and consistency between angle sequences [WW94].

**Invariants of Mating** It is well known that translating one polygon relative to the other does not ruin a valid interpolating surface between them. Several works employ scaling (or “normalization”) as a preprocessing step. Let $A$ and $B$ be the two polygons. Scale $A$ and $B$ in the $x$ and $y$ directions so as to make their bounding axis-parallel rectangles homothetic squares, enclosing the scaled polygons $A'$ and $B'$. Now apply any interpolation algorithm between $A'$ and $B'$. Finally, scale back the polygons and their interpolating surface to their original sizes. This method is used in [CS78, GD82, SH81], to mention just a few. However, there is a flaw in this natural scaling heuristic. O’Rourke [Or94] showed an example in which the tiles forming the connection between $A'$ and $B'$ do not intersect, but the scaled-back triangles do. He also showed that uniform scaling (shrinking or magnifying by the same factor in both $x$ and $y$ directions) does not possess this problem. We show that mating is sensitive to rotation, that is, a pair of matable polygons may become nonmatable if one of them is rotated relative to the other.

**Sufficient Conditions for Matability** Given a pair of polygons, the problem of deciding whether or not these polygons are matable is hard. Our goal is to find conditions that will help us decide if the polygons are matable. In this thesis we present several classes of polygons that can always mate. The fact that the existence of mating is sensitive to the relative rotation of the polygons led us to also look for a condition that considers not only the polygons, but also their relative orientation.

**Mating with Steiner Vertices** A Steiner vertex is a new vertex that is added on the boundary of a polygon. Geiger [Ge93] proved that adding two Steiner vertices on the boundary of the polygons, one on each polygon, allows any pair of polygons to
mate. Since then there have been unsuccessful attempts to show that the addition of just one Steiner vertex, at some carefully-chosen location, allows any pair of polygons to mate. We show that if one polygon is convex, then only one Steiner vertex suffices. (If the two polygons are convex, then no Steiner vertices are needed.)

**Polygon Interpolation by Straight Skeletons** We present an efficient method for interpolating a piecewise-linear surface between two parallel slices, each consisting of one polygon, that define ‘material’ and ‘nonmaterial’ regions. (The algorithm can actually handle slices which contain multiple polygons.) The algorithm is guaranteed to interpolate a valid surface for any possible input, and is intuitive in the sense that it tends, *in practice*, to minimize the surface area of the reconstruction. This is because it uses an offset distance function to locally decide which contour features to bind. The algorithm analyzes the overlay of a pair of slices in order to identify sets of contour portions (bounding a subset of the set of cells of the arrangement of contours) which are to be bound together. Then, the straight skeleton (a linearized version of the medial axis [AAAG95]) of each one of these cells is computed and used to guide a Steiner triangulation of each face of the skeletal cells. Finally, the topology of the skeleton is used again for lifting the triangulation up to three dimensions. The union of the lifted-up triangulations of all the chosen cells is the output surface. We emphasize that the algorithm is fully automatic without any tuning parameters, which are a major disadvantage of many previously-suggested algorithms.

**Organization of the thesis** In chapter 2 we present some definitions and previous works on which we base our new results. In chapter 3 we introduce a few classes of polygons whose matability is guaranteed regardless of their relative position or orientation. In chapter 4 we give a new example of a pair of nonmatable polygons, which is simpler than the one in [GOS96]. In addition, we provide a family of polygon
pairs which are not matable. Chapter 5 presents an efficient interpolation algorithm that uses straight skeletons. We end in chapter 6 with some concluding remarks. As part of this research we developed and implemented an algorithm for matability verification. The software package is described in appendix A.
Chapter 2

Preliminary Results

A polygon \( P \) is given as a sequence of vertices (and the implied edges), so that \( x \in P \) means that \( x \) is on the boundary of \( P \) and not inside it. Throughout this work we denote by \( P \) and \( Q \) (of complexity \( n \) and \( m \), respectively) the polygons that lie in the distinct and parallel planes \( \Pi_P \) and \( \Pi_Q \), respectively. The vertices of \( P \) are \( p_0, \ldots, p_{n-1} \), and the vertices of \( Q \) are \( q_0, \ldots, q_{m-1} \), specified both in counter-clockwise order. A triangle is represented either by its three vertices or by its base edge and its apex vertex.

A connection \( C \) between two polygons \( P \) and \( Q \) is a mapping of any edge of \( P \) to a unique vertex of \( Q \), and any edge of \( Q \) to a unique vertex of \( P \). Since each such edge-vertex connection defines a triangle, this is actually a collection of triangles. Gitlin et al. [GOS96] defined a mating as a connection of \( P \) (which can be either a point, segment, or polygon) and a polygon \( Q \), that forms a triangulated polyhedron (except \( P \) and \( Q \)), that is, a collection of non-intersecting triangles that form a surface homeomorphic to a sphere. Two triangles are said to be non-intersecting if their interiors are disjoint, or they share exactly one vertex or one edge. Every mating between two polygons determines a connection, but naturally not every connection is
a mating. In fact, most connections do not form polyhedra.

In any mating of $P$ and $Q$, each edge of $Q$ is mapped to a unique vertex of $P$, but not every vertex of $P$ is necessarily mapped to by any edge of $Q$. Call a vertex of $P$ to which at least one edge of $Q$ is mapped a transition vertex. For a connection $C$ between $P$ and $Q$, label each vertex of $P$ with the labels of all the vertices of $Q$ to which it is connected (again, in a counter-clockwise order). In Figure 2.1 the vertex $p_0 \in P$ is a transition vertex relative to the edges $q_0q_1, q_1q_2$, and is labeled as $(q_0q_1q_2)$. All labels encountered in a traversal of $P$ form a cyclic sequence of symbols of $q_0, \ldots, q_{m-1}$, so-called a label sequence. Let $s^+$ be a string of one or more occurrences of the symbol $s$.

**Lemma 2.1** [FKU77, Theorem 1] Let $P$ and $Q$ mate by a connection $C$. Then, the associated label sequence must have the regular form $q_i^+, q_{i+1}^+, \ldots, q_{i+m-1}^+$ for some $0 \leq i \leq m$ (where indices are taken modulo $m$), that is, one or more occurrence of $q_i$, followed by one or more occurrence of $q_{i+1}$, and so on.

Note that the label sequence cannot be ordered in the reverse order, $q_{m-1}^+, \ldots, q_0^+$, otherwise the surface will necessarily intersect itself.

In a mating of $P$ and $Q$, each vertex $p_i \in P$ is connected to some vertices $q_j \in Q$. 

Figure 2.1: Label sequence
Each such point-point connection sets constraints on other possible connections, since they cannot intersect the segment $\overline{p_i q_j}$. Consider, for example, a vertex $q' \in Q$. There is an unbounded region $R$ in the plane $\Pi_P$, supporting $P$, such that connecting any point in $R$ with $q'$ will intersect the segment $\overline{p_i q_j}$. Therefore, if there exists a vertex of $P$ in $R$, it cannot be connected to $q'$ in the given mating. Gitlin et al. [GOS96] formulated these constraints by defining “shadows” on $\Pi_P$ of forbidden vertices of $P$. The shadow of $q'$ with respect to $\overline{p_i q_j}$ is a ray that is the locus of all points in the plane $\Pi_P$, such that for any point $p' \in R$, $\overline{p' q'}$ and $\overline{p_i q_j}$ intersect; see Figure 2.2. The shadow of $q'$ with respect to $\overline{p_i q_j}$ is created by “pushing” the point $p_i$ in the direction $\overrightarrow{q_j q_j}$. Any mating of $P$ and $Q$ is a collection of triangles, hence, it will be more convenient to refer to the shadow of a vertex in $Q$ with respect to a triangle of the mating (or of the connection). There are two types of triangles in a mating of $P$ and $Q$: Triangles whose base edge belongs to $Q$ and their apex vertex is in $P$, and similarly, triangles whose base is in $P$ and their apex is in $Q$. Given a vertex $q' \in Q$, the shadow of $q'$ with respect to the triangle $\triangle q_j p_i p_{i+1}$ is a strip bounded by $\overline{p_j p_{i+1}}$ and the ray shadows of $q'$ with respect to $\overline{q_j p_i}$ and $\overline{q_j p_{i+1}}$; see Figure 2.3(a). Therefore, the shadow of $q'$ with respect to the triangle $\triangle q_j p_i p_{i+1}$ is created by pushing the edge $\overline{p_j p_{i+1}}$ along the direction $\overrightarrow{q_j q_j}$. If $\triangle p_i q_j q_{j+1}$ is a triangle of the mating, then the shadow of $q'$ with
(a) The strip shadow of $q'$ with respect to the triangle $\triangle q_j q_i q_{i+1}$

(b) The cone shadow of $q'$ with respect to the triangle $\triangle p_i q_j q_{j+1}$

Figure 2.3: Strip and cone shadows
respect to $\triangle p_i q_j q_j+1$ is the region bounded by the ray shadows of $q'$ with respect to $p_i q_j$ and $p_i q_j+1$; see Figure 2.3(b).

The shadow of $q'$ is denoted a !$q'$ region, where the “!” is to be read as “not,” since $q'$ cannot be connected to any vertex of $P$ that is in this shadow without causing an intersection. Note that there are in general many !$q$ shadows, one for each triangle of the connection $C$.

A major tool in this thesis is a connection obtained by connecting all vertices of $P$ to a unique vertex of $Q$, and all vertices of $Q$ to a unique vertex of $P$. If all the triangles of such a connection are non-intersecting, we call it a double-cone connection between $P$ and $Q$.

**Lemma 2.2** The connection of $P$ and $Q$, obtained by connecting all edges of $P$ to the leftmost vertex of $Q$, and all edges of $Q$ to the rightmost vertex of $P$, is a double-cone (see Figure 2.4).

**Proof.** We denote by $q_\ell$ and $p_r$ the leftmost and rightmost vertices of $Q$ and $P$, respectively, relative to some direction $\overrightarrow{d}$. By connecting each edge of $P$ with $q_\ell$, and
each edge of $Q$ with $p_r$, we obtain two cones having the edge $\overrightarrow{prq}$ in common. By Theorem 1 of [GOS96], any polygon mates with a point. Therefore, triangles of one cone do not intersect each other. We claim that the plane $H$, passing through the edge $\overrightarrow{prq}$ and orthogonal to $\overrightarrow{d}$ separates between the two cones and, hence, there is no intersection of triangles of the two cones. Clearly, the two cones cannot be on the same side of $H$, because $H$ separates between $P$ and $Q$. Therefore, suppose to the contrary that some triangle $(p,e)$ of the connection (where $p \in P$ is a vertex and $e \in Q$ is an edge) intersects that plane $H$. The polygon $Q$ lies below $H$, hence, $p$ must be above $H$. Note that a vertex of $P$ is above $H$ if it lies to the left of the line that is the intersection of $\Pi_P$ and $H$. Since that line passes through $p_r$, a vertex of $P$ is above $H$ if it lies to the left of the vertex $p_r$. Consequently, $p_i$ is not the rightmost vertex of $P$ (with respect to the direction $\overrightarrow{d}$) which is a contradiction to the initial assumption.

\begin{corollary}
The triangles of the connection of $L_P$ and $L_Q$, obtained by connecting every edge of $L_P$ to the leftmost vertex of $L_Q$, and every edge of $L_Q$ to the rightmost vertex of $L_P$, do not intersect each other.
\end{corollary}
Chapter 3

Sufficient Conditions for Matability

Given two parallel planes \( \Pi_P \) and \( \Pi_Q \) and two polygons \( P \in \Pi_P \) and \( Q \in \Pi_Q \), we consider the following question: Under which conditions it is always possible to connect the vertices of \( P \) to the vertices of \( Q \) so as to obtain a mating? The problem of deciding whether or not two polygons are matable is hard. In this chapter we give some simple conditions that let us determine that two polygons are matable.

3.1 Natural Classes of Matable Polygons

A class of matable polygons is a set in which every pair of polygons is matable. For example, any two convex polygons always mate. In this chapter we present two classes of matable polygons. First, we present the class of star-shaped polygons, proving constructively that any two such polygons always mate. Next, we present the class of three-partition polygons, proving, again, the matability of any pair of such polygons.
3.1.1 Star-shaped polygons

**Theorem 3.1** Any pair of star-shaped polygons is matable.

**Proof.** Let $P$ and $Q$ be two star-shaped polygons with points $o_P \in \Pi_P$ and $o_Q \in \Pi_Q$ in the kernels of $P$ and $Q$, respectively, where $\Pi_P$ and $\Pi_Q$ are the respective supporting planes of the two polygons. Since mating is insensitive to $xy$-translation of one polygon relative to the other, we can assume without loss of generality that the points $o_P$ and $o_Q$ are vertically aligned, and denote the middle point of the segment $o_Po_Q$ by $o$ (see Figure 3.1(a)). For every vertex $p_i \in P$ (resp., $q_j \in Q$) we denote the angle between $o_Pp_i$ (resp., $o_Qq_j$) and the $x$ axis by $\angle p_i$ (resp., $\angle q_j$). We sort all vertices of $P$ and $Q$ according to these angles and form a cyclic queue. Without loss of generality, we also assume that the first two vertices in the queue belong to the two polygons, $P$ and $Q$. (We can always rotate the plane to achieve this situation.) Hence, we can denote the first two vertices in the queue as $p_0$ and $q_0$. The first triangle in the reconstruction is defined by the first three vertices in the queue. Now, we proceed in the following manner: If the next vertex belongs to $P$, then we connect it by an edge to the last processed (popped from the queue) vertex of $Q$, and, similarly, if the next vertex belongs to $Q$, then we connect it to the last processed vertex of $P$. (This operation always creates a new triangle in the reconstructed surface.) Let $p_{\text{last}}$ and $q_{\text{last}}$ be the last popped vertices of $P$ and $Q$, respectively.

**Lemma 3.2** The next created triangle dose not intersect the triangles already created in the above process.

**Proof.** Assume without loss of generality that $\angle p_{\text{last}} \geq \angle q_{\text{last}}$. Denote by $S$ the spatial sector containing all the triangles created so far in this process. Formally, $S$ is bounded on one side by the two triangles $\triangle q_0o_Oo_P$ and $\triangle q_0p_qo_O$, and by the extension of the latter triangle over the edge $p_{\text{last}}q_{\text{last}}$ bounded by $\Pi_P$ and $\Pi_Q$ from
Figure 3.1: Starshape polygons always mate

above and below, respectively. Similarly, $S$ is bounded on the other side by the two triangles $\triangle o_q q_{\text{last}} o_P$ and $\triangle p_{\text{last}} q_{\text{last}} o_P$, and by a similar extension of the latter triangle. (See Figure 3.1(b).) Assume first that the next vertex in the queue belongs to $P$. This vertex, $p_{\text{last}+1}$, cannot be inside the sector $S$. Otherwise, we conclude that $\angle p_{\text{last}+1} \leq \angle p_{\text{last}}$, which stands in contradiction to the position of $p_{\text{last}+1}$ in the queue. So $p_{\text{last}+1}$ is outside the sector $S$. The new triangle $\triangle p_{\text{last}} q_{\text{last}} p_{\text{last}+1}$ does not intersect the triangle $\triangle p_{\text{last}} q_{\text{last}} o_P$ because these two triangles share the edge $\overline{p_{\text{last}} q_{\text{last}}}$. Furthermore, the triangle $\triangle p_{\text{last}} q_{\text{last}} p_{\text{last}+1}$ lies outside $S$ since so does $p_{\text{last}+1}$. In addition, the new triangle cannot intersect the triangle $\triangle o_q q_{\text{last}} o_P$ because they share only one vertex, $q_{\text{last}}$, and the other two vertices of the former triangle lie within one side of the plane supporting the latter triangle.

Now assume that the next vertex in the queue belongs to $Q$. Since $\angle q_{\text{last}+1} \geq \angle p_{\text{last}}$, this case is even easier than the previous case, and similar arguments show that the new triangle must be outside $S$. 
In summary, unless the tiling “wraps around,” the new triangle cannot intersect all the triangles previously constructed in this process.

The only situation in which the triangle \( \triangle p_{\text{last+1}}p_{\text{last}}q_{\text{last}} \) intersects the sector \( S \) is when \( \angle p_{\text{last}}o_p p_{\text{last+1}} \geq \pi \), in which case the kernel point \( o_p \) is not in the interior of \( P \), which is a contradiction.

The case in which \( \angle q_{\text{last}} > \angle p_{\text{last}} \) is completely symmetric: we just need to choose the other “diagonal,” \( o_p q_{\text{last}} \).

\[ \square \]

Denote by \( C \) the collection of all the triangles created during the process described above. Lemma 3.2 guarantees that all the triangles in \( C \) are pair-wise disjoint. In addition, all the triangles in \( C \) are defined over the vertices of the polygons \( P \) and \( Q \). Hence, there is no intersection between the triangles of \( C \) and the polygons \( P, Q \). The reconstructed polyhedron is then bounded by \( P, Q, \) and all the triangles of \( C \). \( \square \)

### 3.1.2 3-partition polygons

Given a simple polygon \( P \) we define a partition of \( P \). The partition point of \( P \) is a point, \( p \), in the interior of \( P \), such that there are rays emanating from \( p \), intersecting the polygon only once, and the intersection points of the rays with \( P \) are vertices of the polygon \( P \). We denote those rays and vertices by \( r_k^P \) and \( p_{i_k} \) respectively. We also denote the polygonal line, obtained by all vertices of \( P \) between \( p_{i_k} \) and \( p_{i_{k+1}} \) (counter clock wise), by \( PL_{k,k+1} \). Denote the sector between \( r_k^P \) and \( r_{k+1}^P \) by \( s_k^P \) (\( s_k^P \) is the polygon \( p, p_{i_k}, PL_{k,k+1}, p_{i_{k+1}} \)). Here, we will restrict our discussion to polygons which has a partition point with three such rays; see Figure 3.2.

**Definition 3.3** A polygon \( P \) has a 3-partition if it contains a partition point \( p \) and three of its vertices, \( p_{i_1}, p_{i_2}, p_{i_3} \), fulfill

18
Figure 3.2: A 3-partition polygon

1. The rays emanating from \( p \) towards \( p_{ik} \) (for \( k = 1, 2, 3 \)) intersect \( P \) only once;

2. The angle between two successive rays is less than \( \pi \); and

3. The distance between \( p \) and \( p_{ik} \) is greater than or equal to the distances between \( p \) and the vertices in the polygonal lines \( PL_{k-1,k} \) and \( PL_{k,k+1} \).

Hereafter we call these polygons “3-partition polygons.”

**Theorem 3.4** Any pair of 3-partition polygons is matable.

**Proof.**

We prove the claim constructively by using the following mating algorithm. Split each of the two polygons into three polygons (the closed sectors) using the partitioning rays. Next, match each of the three subpolygons of \( P \) to a unique subpolygon of \( Q \).
in order to obtain three pairs of polygons. (We show below how we do this.) Then, mate each couple of subpolygons separately. Finally, combine the three polyhedra by canceling out duplicate triangles.

In order to match the sectors of $P$ with the sectors of $Q$ we compute the overlay of the two polygons obtained by projecting one slice onto the other. Since we may assume without loss of generality that the partition points $p$ and $q$ are vertically aligned, these two points are unified in the overlay. The ray $r^P_k$ is matched with the ray of $Q$ that forms with it the smallest angle. The chosen ray is fixed as $r^Q_k$, and the other two rays $r^Q_2$ and $r^Q_3$ follow it in a counter-clockwise order. Consider the three pairs of sectors. In the projection of one slice onto the other, there are at most two rays of $P$ in each sector of $Q$, and vice versa. Thus, the overlay of any couple of sectors, $s^P_k$ and $s^Q_k$, has one of two forms: either each one of the sectors contains one ray of the other sector (see Figure 3.3(a)), or one sector is fully contained in the other (see Figure 3.3(b)).

**Lemma 3.5** Each pair of matched sectors $s^P_k, s^Q_k$ is matable.

**Proof.** We begin the construction of a mating of $s^P_k$ and $s^Q_k$ we compute the cross product of $r^P_k$ and $r^Q_k$. This is done in order to find whether $r^Q_k$ is rotationally on the left or on the right of $r^P_k$. If $r^P_k \times r^Q_k \geq 0$, we connect all the vertices of $s^P_k$ (except $p$) to $q_{ik}$, and all the vertices of $s^Q_k$ (except $q$) to the vertex $p_{ik+1}$. The resulting collection
of triangles forms two non-intersecting fans having the edge $p_{i_{k+1}}q_{j_k}$ in common; see Figure 3.4. Let us argue why there are no intersections. Triangles within each fan do not intersect one another (cf. [GOS96, Theorem 1]). Definition 3.3 guarantees the existence of a direction in which the two points $p_{i_{k+1}}$ and $q_{j_k}$ are the leftmost and the rightmost of their polygons (see Figure 3.5). Thus, according to Lemma 2.2, there is a plane separating the two fans. Therefore the two fans do not intersect. Similarly, if $r^P_k \times r^Q_k < 0$, we connect all the vertices of $s^P_k$ (except $p$) to $q_{j_{k+1}}$, and all the vertices of $s^Q_k$ (except $q$) to $p_{i_k}$. Again, we obtain two non-intersecting fans having the edge
in common. Now, it remains to connect $p$ to $q$, $q_{jK}$, and $q_{jK+1}$ in order to get a mating of the two sectors.  

Hence, each of the three pairs of sectors mate. Now we need to "glue" the three resulting polyhedra. We glue polyhedra on the common triangles $\triangle pq_{j1}q$, $\triangle pq_{j2}q$, and $\triangle p, q_{j3}, q$. By removing the duplicate triangles we obtain a mating of the two polygons.  

Note that it is sufficient to require, for two matched sectors $s_P^k$ and $s_Q^k$, the existence of a direction along which $p_{ik}$ is the leftmost vertex of $P$ and $q_{ik+1}$ is the rightmost vertex of $Q$. The third condition of the 3-partition definition is used for constructing a mating of a pair of matched sectors. Indeed, if $p_{ik}$ is the farthest from $p$ among all the vertices in the sectors defined by $p_{ik}$, then this weaker condition is also satisfied. However, we use the original version of the third condition (although it is stronger) because it imposes a requirement on the polygons and not on their relative orientation.

### 3.2 Pairs of Matable Polygons

The fact that the matability of two polygons $P$ and $Q$ is sensitive to rotation led us to look for conditions that consider the relative orientation of the polygons. We prove the conditions that we found in a constructive manner, by connecting all the vertices of $P$ to the leftmost vertex of $Q$ and all the vertices of $Q$ to the rightmost vertex of $P$, obtaining two "cones" that share one edge and are separable by a plain. Then, a mating is obtained by a small refinement of the connection.

**Definition 3.6** Two simple polygons $P$ and $Q$ are parallel-separable if there exist two vertices $p_i \in P, q_j \in Q$, such that the following holds:

1. There is a direction along which $p_i$ is the rightmost vertex of $P$ and $q_j$ is the
leftmost vertex of \( Q \);

2. There is no vertex of \( P \) in the shadow of \( q_j \) with respect to the triangle \( \triangle p_ip_{i-1}q_{j+1} \);

and

3. There is no vertex of \( Q \) in the shadow of \( p_i \) with respect to the triangle \( \triangle q_jq_{j+1}p_{i-1} \).

See Figure 3.6 for an illustration.

**Theorem 3.7** Any pair of parallel-separable polygons is matable.

**Proof.** Let \( P \) and \( Q \) be two parallel-separable polygons in the planes \( \Pi_P \) and \( \Pi_Q \). We begin by connecting each vertex of \( P \) to the leftmost vertex \( q_j \in Q \), and each vertex of \( Q \) to the rightmost vertex \( p_i \in P \), obtaining two non-intersecting polyhedral cones having the edge \( p_ip_j \) in common (see Figure 3.7). By Lemma 2.2 we know that the two cones are separable by a plane passing through \( p_i \) and \( q_j \). The cone \((P,q_j)\) lies above that plane, while the cone \((Q,p_i)\) lies below that plane. Let \( C \) be the union of the two cones. Although the triangles of \( C \) do not intersect one another, the connection \( C \) is not a mating since the closed polyhedron is not simple.

Consider the tetrahedron \( \{q_j, q_{j+1}, p_i, p_{i+1}\} \). Since the shadows of \( q_j \) and \( p_i \) with respect to the triangles \( \triangle p_{i-1}p_ip_j \) and \( \triangle q_jq_{j+1}p_{i-1} \), respectively, are empty, no edge of \( C \) intersects with or is contained in this tetrahedron. The edge \( p_{i-1}q_{j+1} \) lies completely outside \( C \). Therefore, we can replace in \( C \) the triangles \( \triangle p_{i-1}p_ip_j \) and \( \triangle q_{j+1}q_{j}p_i \) by \( \triangle p_{i-1}q_{j}p_i \) and \( \triangle q_jq_{j+1}p_{i-1} \). In other words, we open the fold between the triangles \( \triangle p_{i-1}p_ip_j \) and \( \triangle q_jq_{j+1}q_{j}p_i \) by introducing the tetrahedron \( \{p_{i-1}, p_i, q_{j+1}, q_j\} \). Formally, the modified connection \( C' \) is defined as \( (\text{Cone}(P,q_j) \setminus \triangle p_{i-1}p_{j}q_{j}) \cup (\text{Cone}(Q,p_i) \setminus \triangle q_{j+1}q_{j}p_i) \cup \triangle q_{j+1}p_{i-1}q_{j} \cup \triangle q_{j}q_{j+1}p_{i-1} \), which is a mating of \( P \) and \( Q \).

Note that the same proof holds if we take the vertices \( p_i, p_{i-1} \) of \( P \) and the vertices \( q_{j-1}, q_j \in Q \).
(a) $p_i$ is the rightmost vertex of $P$ and $q_j$ is the leftmost vertex of $Q$

(b) The shadow of $q_j$ with respect to the triangle $\triangle p_i p_{i-1} q_{j+1}$ is empty

(c) The shadow of $p_i$ with respect to the triangle $\triangle p_{i-1} q_j q_{j+1}$ is empty

Figure 3.6: Parallel-separable polygons
We now present two simpler and intuitive conditions which are derived from Theorem 3.7.

**Theorem 3.8** Given two simple polygons $P$ and $Q$, if there are two vertices $p_i \in P, q_j \in Q$, such that the following holds:

1. There is a direction along which $p_i$ and $q_j$ are the extreme vertices of $P$ and $Q$, respectively, in opposite sides; and

2. $p_{i-1}, p_i, p_{i+1} \in CH(P)$ and $q_{j-1}, q_j, q_{j+1} \in CH(Q)$;

then $P$ and $Q$ are matable.

**Proof.** We begin by connecting each vertex of $P$ with the extreme (so-called leftmost) vertex $q_j \in Q$, and each vertex of $Q$ with the extreme (so-called rightmost) vertex $p_i \in P$. By Lemma 2.2 we know that this connection is a double-cone made of two non-intersecting cones having the edge $p_i q_j$ in common (see Figure 3.8). Now we
Figure 3.8: A double-cone between $P$ and $Q$
consider the edges of $P$ and $Q$ incident to $p_i$ and $q_j$, respectively. We project the edges $q_jq_{j+1}$ and $q_{j-1}q_j$ along the line $pq_j$ onto the plane $\Pi_P$, obtaining the segments $q_jq_{j+1}$ and $q_{j-1}q_j$ (see Figure 3.9 (a)). We denote some angles as $\alpha = \angle p_{i-1}p_ip_{i+1}$, $\beta = \angle q_jq_{j-1}q_{j+1}$, $\gamma = \angle q_{j-1}p_ip_{i+1}$, and $\delta = \angle p_{i-1}p_ip_{i+1}$ (see Figure 3.9 (b)). Trivially, one of the angles $\gamma$ and $\delta$ has to be smaller than $\pi$. Assume without loss of generality that $\delta < \pi$. The region bounded by $p_{i-1}p_i$ and the two rays emanating from $p_{i-1}$ and $p_i$ in the direction $q_jq_{j+1}$ is the shadow of $q_j$ with respect to the triangle $\triangle p_{i-1}p_ip_{i+1}$; see Figure 3.10. Likewise, the shadow of $p_i$ with respect to the triangle $\triangle q_jq_{j+1}p_{i-1}$ is the region bounded by $q_jq_{j+1}$ and the two rays emanating from $q_j$ and $q_{j+1}$ in the direction $p_ip_{i-1}$.

Since $p_{i-1}, p_i, p_{i+1} \in \text{CH}(P)$, the edge $p_{i-1}p_i$ is an edge of $\text{CH}(P)$. Therefore, all the vertices of $P$ are on the same side of the line defined by $p_{i-1}$ and $p_i$. Hence, the shadow of $q_j$ with respect to the triangle $\triangle p_{i-1}p_ip_{i+1}$ contains no vertices of $P$. Similarly, the shadow of $q_j$ with respect to the triangle $\triangle q_jq_{j+1}p_{i-1}$ does not contain any vertex of $P$. 

Figure 3.9: Building the shadow
Figure 3.10: The shadows of $p_i$ and $q_j$ with respect to $\Delta q_j q_{j+1} p_{i-1}$ and $\Delta p_i p_{i-1} q_{j+1}$, respectively
Therefore, \( P \) and \( Q \) are parallel-separable and hence, by Theorem 3.7, matable. \( \square \)

**Theorem 3.9** Two simple polygons \( P \) and \( Q \) are matable if there are two vertices \( p_i \in P, q_j \in Q \) such that the following holds:

1. There is a direction along which \( p_i \) and \( q_j \) are the extreme vertices of \( P \) and \( Q \), respectively, in opposite sides;
2. \( p_{i-1}, p_i \in CH(P) \) and \( q_j, q_{j+1} \in CH(Q) \); and
3. The angle between \( \overrightarrow{p_{i-1}p_i} \) and the projection of \( \overrightarrow{q_jq_{j+1}} \) on \( \Pi_P \) along the line \( \overrightarrow{p_iq_j} \) is less than \( \pi \).

**Proof.** Similarly to the proof of Theorem 3.8, we begin by connecting each vertex of \( P \) with the extreme (so-called leftmost) vertex \( q_j \in Q \), and each vertex of \( Q \) with the extreme (so-called rightmost) vertex \( p_i \in P \), obtaining two non-intersecting cones having the edge \( \overrightarrow{p_{i-1}p_i} \) in common. Since the edge \( \overrightarrow{p_{i-1}p_i} \) is an edge of \( CH(P) \), all the vertices of \( P \) must be on the same side of the line defined by \( p_{i-1} \) and \( p_i \). Hence, the shadow of \( q_j \) with respect to the triangle \( \triangle p_{i-1}p_iq_{j+1} \) contains no vertices of \( P \). Similarly, the edge \( \overrightarrow{p_{i-1}p_i} \) is an edge of \( CH(P) \), and hence all the vertices of \( P \) must be on the same side of the line defined by \( p_{i-1} \) and \( p_i \). Therefore, the shadow of \( q_j \) with respect to the triangle \( \triangle p_{i-1}q_jq_{j+1} \) contains no vertex of \( Q \). Consequently, the polygons \( P \) and \( Q \) are parallel separable, and hence, by Theorem 3.7, matable. \( \square \)
Chapter 4

Nonmatable Polygons

Although there has been considerable research on reconstructing three-dimensional objects from parallel slices, only in 1996 Gitlin et al. [GOS96] raised the question of whether the reconstruction is always possible. They presented a pair of nonmatable polygons consisting of a 63-vertex polygon $P$ and a triangle $Q$. Instead of proving the nonmatability of $P$ and $Q$, they proved the nonmatability of $Q$ and a polygon $P'$ which is much more complex than $P$, and then argued that it is easy to extend the proof for $P$. In this chapter we give an example simpler than that of [GOS96]: A 45-vertex polygon $P$ and a triangle $Q$. We provide a complete proof of the nonmatability of this pair, which is currently the least-complexity known example of nonmatable polygons, and the only fully-proven such example. We also provide, for the first time, a family of nonmatable pairs of polygons with unbounded complexity. In Section 4.1 we give a general description of the two polygons; In Section 4.2 we provide some properties and observations about the polygons; In Section 4.3 we prove their nonmatability; and in Section 4.4 we introduce the family of nonmatable pairs of polygons.
4.1 General Description

In this section we provide an example of a pair of polygons $P$ and $Q$ that cannot mate. The polygon $P = (p_0, \ldots, p_{44})$, shown in Figure 4.1, is a complex polygon with 45 vertices, while $Q = (q_0, q_1, q_2)$ is a triangle. The edges of $P$ consist of three groups: spikes, bays, and anvils (denoted as $S_i$, $B_j$, and $A_k$, respectively). There are three spikes, each consisting of two edges, with spike tips at $p_0$, $p_{15}$, and $p_{30}$. Thus, $S_0 = (p_{44}, p_0, p_1)$, $S_1 = (p_{14}, p_{15}, p_{16})$, and $S_2 = (p_{29}, p_{30}, p_{31})$. There are three anvils, each consisting of three edges: $A_0 = (p_7, p_8, p_{10})$, $A_1 = (p_{22}, p_{23}, p_{24}, p_{25})$, and $A_2 = (p_{37}, p_{38}, p_{39}, p_{40})$. In addition, there are six bays. The first two bays are $B_0 = (p_1, \ldots, p_7)$, $B_1 = (p_{10}, \ldots, p_{14})$, and the other four bays $B_2, \ldots, B_5$ are defined in a similar manner. See Figure 4.1. The polygon $Q$ has coordinates $(1543,1208)$, $(2914,2000)$, and $(1543,2791)$, and the actual coordinates of the vertices of $P$ are listed in Table 4.1.

4.2 Properties

We now provide a few observations about the polygons $P$ and $Q$.

Lemma 4.1 Let $C$ be a connection between $P$ and $Q$. If there exists a vertex $q \in Q$ such that every bay of $P$ contains a vertex connected to $q$, then $P$ and $Q$ do not mate by $C$.

Proof. Because of the rotational symmetry of the entire configuration, we can assume without loss of generality that $q = q_0$. If every bay of $P$ has an edge labeled $q_0$, then all three transition vertices are on or between two consecutive bays $B_i, B_{i+1}$. Consequently, if every bay of $P$ has an edge labeled $q_0$, then at most one anvil (or one spike) may have a label that is not $q_0$. The remainder of the proof falls into three
Figure 4.1: The 45-vertex polygon $P$
similar cases, depending on whether the transition vertex of \((q_1, q_2)\) is on a bay, an anvil, or a spike.

1. The transition vertex is on a bay.

   Every possible location of the transition vertex \(p_i\) on any causes the shadow \(!q_0\) with respect to \(\triangle p_iq_1q_2\) to cover vertices of both an anvil and a spike, so there are an anvil and a spike with at least one vertex that is not labeled \(q_0\). This violates the fact that at most one anvil (or one spike) may have a label that is not \(q_0\). For example, if the transition vertex were \(p_7\), then the \(!q_0\) shadow would cover both \(S_0\) and \(A_0\), as shown in Figure 4.2.

2. The transition vertex is on an anvil.

   The vertex must be an endpoint of a base edge of an anvil, otherwise we fall into the first case. However as is clear from Figure 4.3, the tip of one of the
spikes will fall into the \( q_0 \) shadow, and again we will obtain a contradiction.

3. The transition vertex is on a spike.

The vertex must be the tip of a spike, otherwise we fall into the first case. Assume that \((p_0, q_1, q_2)\) is part of the mating. The vertex \( p_0 \) cannot serve as all three transition points since connecting all three vertices of \( Q \) to \( p_0 \) will yield a double-cone and not a valid polyhedron. Hence, one of the two vertices adjacent to \( p_0 \) is not labeled \( q_0 \). Assume that \( p_1 \) is connected to \( q_2 \). (A similar argument holds for the case in which \( p_{44} \) is connected to \( q_1 \).) The connection of the edge \( p_0 p_1 \) to \( q_1 \) casts a \( q_0 \) strip shadow parallel to \( q_0 q_2 \). As shown in Figure 4.4, this strip intersects the anvil \( A_0 \) and forces \( A_0 \) to contain a label that is not \( q_0 \). This violates the fact that at most one anvil (or one spike) may have a label that is not \( q_0 \). The case in which the transition vertex is \( p_{15} \), or, symmetrically, \( p_{30} \), is denied in exactly the same fashion.
Figure 4.3: The transition vertex between \((q_1, q_2)\) is on an anvil (Lemma 4.1)

Figure 4.4: The transition vertex between \((q_1, q_2)\) is on a spike (Lemma 4.1)
Lemma 4.2 Let $C$ be a connection between $P$ and $Q$. If every vertex of the bay $B_0 \in P$ is connected to the vertex $q_0 \in Q$, then $P$ and $Q$ cannot mate by $C$.

Proof. The connection of every vertex of $B_0$ to $q_0$ casts a $!q_1$ strip shadow, whose bounding rays are parallel to $\overline{q_1q_0}$. This connection also casts a $!q_2$ shadow, whose bounding rays are parallel to $\overline{q_2q_0}$ (see Figure 4.5 for an illustration). The intersection of these two shadows is a region where no vertex of $P$ can be connected to neither $q_1$ nor $q_2$, so that all vertices of $P$ in this region, denoted a $q_0$-region, must be connected to $q_0$. Notice that all the six bays of $Q$ intersect the $q_0$-region. Therefore, all six bays must have at least one vertex labeled $q_0$, and hence, by Lemma 4.1, $P$ and $Q$ cannot mate by this connection. \[\square\]
Every anvil base-edge has a vertex in $Q$ which is closest to that anvil. By closest we mean that its Euclidian distance to the anvil is the smallest among the vertices of $Q$. One can observe that the anvil base-edges connections $(p_8, p_9, q_2)$, $(p_{23}, p_{24}, q_0)$, and $(p_{38}, p_{39}, q_1)$, i.e., the connections in which every anvil base-edge is connected to its closest vertex of $Q$, are the most “natural” in the sense that the resulting strip shadows are empty (see Figure 4.6).
Indeed, as we show next, all connections that connect one (or more) anvils base edge to a vertex of $Q$ other than its closest, do not yield a mating. The case in which two anvil base edges are not connected to their nearest vertex of $Q$ is rather simple. Assume without loss of generality that the triangles $\triangle q_0 p_8 p_9$ and $\triangle q_1 p_{23} p_{24}$ are part of the connection. Then, the two triangles cast $!q_2$ and $!q_0$ shadows (see Figure 4.7), and their intersection forms a region in which all vertices are connected to $q_1$, that covers at least one vertex in every bay of $P$. By Lemma 4.1, the connections will not yield a mating. The case where all three base edges are not labeled by their closest vertices of $Q$ falls into the case described above. Let us now consider the case in which only one anvil base is not labeled by its closest vertex of $Q$.

**Lemma 4.3** Let $C$ be a connection between $P$ and $Q$. If one of the three anvil bases is not connected to its nearest vertex of $Q$, then $P$ and $Q$ cannot mate by $C$.

**Proof.** Assume to the contrary that $C$ is a legal mating. We assume without loss of generality (using again the rotational symmetry) that $p_8 p_9$ is not connected to $q_2$. Then, either $\triangle p_8 p_9 q_0$ or $\triangle p_8 p_9 q_1$ is a triangle of the mating.

First, assume that the triangle $\triangle p_8 p_9 q_0$ belongs to the mating. This triangle casts a $!q_2$ shadow shown lightly shaded in Figure 4.8. The vertices of the bay $B_0$ are in the $!q_2$ shadow and cannot be labeled $q_2$. In addition, all the vertices of $B_0$ appear in the associated label sequence before $q_0$. Hence, the label $q_1$ is not possible. Therefore all vertices of the bay $B_0$ are connected to $q_0$. Lemma 4.2 guarantees them that no mating exists.

Second, if the triangle $\triangle p_8 p_9 q_1$ belongs to the mating, then it casts a $!q_2$ shadow shown lightly shaded in Figure 4.9. This $!q_2$ shadow forces the transition vertices $(q_1, q_2)$ and $(q_2, q_0)$ to be on a spike or on an anvil. Therefore, the bay $B_1$ is labeled $q_1$. Each edge of $B_1$ casts a $!q_0$ strip shadow whose bounding rays are parallel to $\overline{q_0 q_1}$. These strips form the dark shaded region in Figure 4.9. The intersection of these
Figure 4.7: The $!q_2$ and $!q_0$ shadows and their intersection, all vertices in the dark grey region are connected to $q_1$. 
Figure 4.8: The $!q_2$ shadow cast by the triangle $\triangle p_8p_9q_0$ (Lemma 4.3)
strips with the $!q_2$ shadow is a region $R$ within which all vertices must connect to $q_1$. Since the bay $B_2$ intersects $R$, and the transition vertices $(q_1, q_2)$ and $(q_2, q_0)$ are on a spike or on an anvil, all the vertices of $B_2$ are labeled $q_1$. The application of Lemma 4.2 concludes the proof.

It is left to show that connecting every anvil to its closest vertex of $Q$ will not yield a mating either. The contradiction will be obtained by showing that connecting every anvil to its closest vertex of $Q$ compels a bay labelling of the form: $B_2$ is labeled $q_0$; $B_4$ is labeled $q_1$; and $B_0$ is labeled $q_2$; which is not possible.
First connect the base of $A_0$ to $q_2$, the base of $A_1$ to $q_0$, and the base of $A_2$ to $q_1$. In order to show that the bay $B_2$ is labeled $q_0$, we show that the transition vertex $(q_2, q_0)$ does not belong neither to $B_2$ nor to $A_1$.

**Lemma 4.4** *The transition vertex $(q_2, q_0)$ cannot be any vertex of $A_1$.*

**Proof.** Since the base of $A_1$ is connected to $q_0$, the two vertices that could be the transition vertex between $q_2$ and $q_0$ are $p_{22}$ and $p_{23}$.

Assume first that $p_{22}$ is the transition vertex. Then, $B_1$ and $B_2$ are labeled $q_2$. The $!q_0$ shadow cast by this connection covers $B_3$, therefore $B_3$ is labeled $q_1$. This is not possible because, as can be seen in Figure 4.10, the tip of the bay $B_3$ is in the $!q_1$ shadow cast by the connection of $B_1$ and $B_2$ to $q_2$.

In case $p_{23}$ is the $(q_2, q_0)$ transition vertex, we have the same shading effect, and the contradiction is obtained by an identical argument. \(\square\)

If $S_1$ is labeled $q_2$, then it casts a $!q_0$ shadow that covers $p_{21}$ (see Figure 4.11) and enforces one of the vertices of $A_1$ to be the transition vertex, which is not possible by Lemma 4.4.

By applying the same arguments for the other two pairs of anvils, we get the following labelling: $B_2$ is labeled $q_0$; $B_4$ is labeled $q_1$; and $B_0$ is labeled $q_2$. However, this is not possible either since the connection of $B_2$ to $q_0$ casts a $!q_1$ shadow on the tip of $B_4$ and a $!q_2$ shadow on the tip of $B_0$ (see Figure 4.12).

This completes the proof of the main theorem:

**Theorem 4.5** *The polygons $P$ and $Q$ are nonmatable.*
Figure 4.10: The $!q_2$ shadow cast by the triangle $\triangle p_8 p_9 q_1$ (Lemma 4.4)
Figure 4.11: The $q_0$ shadow covers the vertex $p_{21}$ and enforces one of the vertices of $A_1$ to be a transition vertex.
Figure 4.12: The $q_2$ shadow cast by the triangle $\triangle p_8p_9q_1$
4.4 A Family of Nonmatable Pairs of Polygons

In this section we present a family of pairs of nonmatable polygons. We describe in detail the construction of each such pair in the family. Each pair consists of a regular polygon with \( n \) vertices and a complex polygon with \( 4n^2 + 9n \) vertices. The pairs were checked by software (in which the algorithm given in Appendix A was implemented) and were found to be nonmatable. The largest pair of polygons checked is of complexity \( n = 10 \), a regular decagon, and its 480-vertex counterpart. The smallest complexity pair is shown in Figure 4.13. Note that the polygon in Figure 4.13 is different than the polygon presented in the previous sections. The former polygon has 45 vertices and was conceived manually. The polygon in Figure 4.13 was generated by the process we describe here, and has 63 vertices.

All 63 vertices of this polygon are placed on 7 concentric circles with growing
radii. The vertices on the inner circle are the tips of the bays, the vertices on the next four circles create the whirlpool of the bays, and the vertices on the two circles with the largest radii form the anvils and the spikes. The general algorithm constructs a polygon with $2n$ bays, $n$ anvils, and $n$ spikes, which cannot mate with a regular polygon with $n$ vertices. We draw $n + 4$ concentric circles with growing radii. Then, we equally spread $2n$ points on the inner-most circle. These $2n$ points will be the tips of $2n$ bays. Now we locate $4n$ points on each of the $n + 1$ circles. This is done in order to create the “whirlpool” around the center of the circles. The next circle will carry the $2n$ vertices of the $n$ anvil-bases, and the last circle will have the $n$ tips of the $n$ spikes. In total, we obtain a polygon with $2n + 4(n + 1) + 3n = 4n^2 + 9n$ vertices, that is nonmatable with a carefully-oriented regular polygon with $n$ vertices. (Figure 4.14 illustrates the algorithm for $n = 4$.)

Given a regular polygon $Q_n$ with $n$ vertices, the following 12-step algorithm constructs a “monster” polygon $P_n$, such that $Q_n$ and $P_n$ are nonmatable.

1. Place the center of the regular polygon bounded in the unit circle at the origin. The unit circle will be labeled $c_0$ (Figure 4.14(a)).

2. Construct $n + 1$ more concentric circles centered at the origin with radii $3, 5, \ldots, 2(n + 1) + 1$
   labeled $c_1, \ldots, c_{n+1}$ (Figure 4.14(b)).

3. Construct $8n$ rays emanating from the origin, such that $n$ rays intersect the vertices of the regular polygon, and all rays divide the circle into $8n$ arcs of identical length. The rays are labeled $r_0, \ldots, r_{8n-1}$ such that the ray labeled $r_0$ intersects the regular polygon at a vertex (Figure 4.14(c)).

4. Place $2n$ vertices at the intersection points of the unit circle $c_0$ and the rays $r_{4j}$, for $0 \leq j \leq 2n - 1$ (Figures 4.14(d,e)).
5. For each circle \( c_i \), \( 1 \leq i \leq n + 1 \), place \( 4n \) vertices at the intersection points of the circle \( c_i \) and the rays \( r_{2i} \), for \( 0 \leq i \leq 4n - 1 \) (Figure 4.14(f)).

6. Label each vertex defined by the intersection of the circle \( c_i \) and the ray \( r_j \) as \( p^i_j \).

7. Connect the vertices to obtain the \( 2n \) bays. The tip of the bay is the vertex \( p^0_j \) on the circle \( c_0 \). Every bay has two banks. Build the first bank by connecting the vertices

\[
p^0_j \rightarrow p^1_j + 4 \rightarrow p^2_j + 8 \rightarrow p^3_j + 4i \rightarrow \cdots \rightarrow p^{n+1}_j + 4(n+1)
\]

and the opposite bank by connecting

\[
p^0_j \rightarrow p^1_{j+2} \rightarrow p^2_{j+2} + 8 \rightarrow p^3_{j+2} + 4i \rightarrow \cdots \rightarrow p^{n+1}_{j+2} + 4(n+1) + 2
\]

(Figures 4.14(g,h)).

8. Draw an additional circle centered at the origin and with radius \( 6n + 1 \), and label it \( c_{n+2} \).

9. Place the \( 2n \) vertices of the anvil-bases at the intersection point of the circle created in step (8) and the rays \( r_{2+8i} \) and \( r_{((2+8i)+7) \mod (8n)} \), for \( 0 \leq i \leq n - 1 \) (Figure 4.14(i)).

10. Build the \( n \) anvils by connecting the vertices

\[
p^{n+1}_j \rightarrow p^{n+2}_j \rightarrow p^{n+2}_{j+7} \rightarrow p^{n+1}_{j+2}
\]

for \( 0 \leq i \leq n - 1 \) and \( j = 2 + 8i \) (Figure 4.14(j)).

11. Let us denote the two side edges of an anvil by \( a_{\text{left}} \) and \( a_{\text{right}} \). Place \( n \) vertices at the intersection of the extensions of the edges \( a_{\text{left}} \) of one anvil and the edge \( a_{\text{right}} \) of its following (counter-clockwise) anvil. Label the \( n \) vertices \( s_i \), for \( 0 \leq i \leq (n - 1) \) (Figure 4.14(k)).

12. Build the spikes by connecting the vertices
\[ p_j^{n+1} \rightarrow s_i \rightarrow p_{j+2}^{n+1} \]

for \( 0 \leq i \leq (n - 1) \) and \( j = 6 + 8i \), (Figures 4.14(l)).

Figure 4.15 shows the pairs polygons generated by the algorithm for \( n = 4, 5, 6, 7 \).
Figure 4.14: Execution of the algorithm
Figure 4.14 (continued)

(e) Close-up of (d)
Figure 4.14 (continued)
Figure 4.14 (continued)
Figure 4.14 (continued)
Figure 4.14 (continued)
Figure 4.15: Family of nonmatable polygons

(a) A square and its respective “monster”

(b) A regular pentagon and its respective “monster”
(c) A regular hexagon and its respective “monster”

(d) A regular septagon and its respective “monster”

Figure 4.15 (continued)
Chapter 5

Polygon Interpolation by Straight Skeletons

5.1 Introduction

In this chapter we consider the matability and interpolation between polygons when the addition of Steiner points are permitted. We first prove a theorem about situations in which one Steiner vertex suffices. Then we present a general interpolation algorithm that makes use of many such vertices.

An interesting question is whether or not a pair of polygons can be nonmatable regardless of how many vertices one sprinkles on their boundaries. Geiger [Ge93] proved that adding two Steiner vertices on the boundary of the polygons, one on each polygon, allows any pair of polygons to mate. In certain circumstances one Steiner vertex suffices.

Theorem 5.1 Two simple polygons $P$ and $Q$, such that $P$ contains three consecutive vertices $p_{i-1}, p_i, p_{i+1} \in CH(P)$, can always be made matable by adding one Steiner
vertex on the boundary of \( Q \).

**Proof.** Refer to Figure 5.1. Consider one of the directions along which \( p_i \) is the rightmost vertex of \( P \), and denote by \( q_j \) the vertex of \( Q \) that is the leftmost along that direction. Consider the edges of \( P \) and \( Q \) incident to \( p_i \) and \( q_j \), respectively. We project the edges \( q_j q_{j+1} \) and \( q_{j-1} q_j \) along the line \( p_i q_j \) onto the plane \( \Pi_P \), obtaining the segments \( \overrightarrow{q_{j+1} p_i} \) and \( \overrightarrow{q_{j-1} p_i} \). Fix the angles \( \alpha = \angle p_{i-1} p_i p_{i+1} \), \( \beta = \angle q'_{j-1} q_j q'_{j+1} \), \( \gamma = \angle q'_{j-1} p_i p_{i+1} \), and \( \delta = \angle p_{i-1} p_i q'_{i+1} \). Trivially, one of the angles \( \gamma \) and \( \delta \) has to be smaller than \( \pi \). Assume without loss of generality that \( \delta < \pi \). The edge \( p_{i-1} p_i \) is an edge of \( \text{CH}(P) \). Therefore, all the vertices of \( P \) are on the same side of the line defined by \( p_{i-1} \) and \( p_i \). Hence, the shadow of \( q_j \) with respect to the triangle \( \triangle p_{i-1} p_i q_{i+1} \) contains no vertices of \( P \). However, the edge \( q_j q_{j+1} \) is not an edge of \( \text{CH}(Q) \). Therefore, there is no guarantee that the shadow of \( q_j \) with respect to the triangle \( \triangle q_j q_{j+1} p_{i-1} \) does not contain any vertex of \( Q \). If the shadow of \( p_i \) with respect to the triangle \( \triangle q_j q_{j+1} p_{i-1} \) contains any other vertices of \( Q \), we add a Steiner vertex \( q_s \) on the line segment \( q_j q_{j+1} \) so that the shadow of \( p_i \) with respect to the triangle \( \triangle q_j q_{i+1} p_{i-1} \) contains no other vertex of \( Q \). Therefore, \( P \) and \( Q \) are parallel separable and hence, by Theorem 3.7, matable. \( \square \)

The main focus of this chapter is on an efficient method for interpolating a piecewise-linear surface between two parallel slices, each consisting of one polygon, which uses Steiner vertices. Our method is fully automatic and is guaranteed to produce non-self-intersecting surfaces in all cases. The method is based on computing cells in the overlay of the slices that form the symmetric difference between them. Then, the straight skeleton (a linearized version of the medial axis [AAAG95]) of each one of these cells is computed and used to guide a Steiner triangulation of each face of the skeletonized cells. Finally, the topology of the skeleton is used again for lifting the triangulation up to three dimensions. The union of the lifted-up triangulations of all the chosen cells is the output surface. The algorithm is fully automatic without
any tuning parameters, which are a major disadvantage of some previously-suggested algorithms. We provide some experimental results on various complex examples to show the good and robust performance of our algorithm. In Section 5.2 we give an overview of the algorithm. Section 5.3 describes the analysis of the overlay of two slices. Section 5.4 describes the computation of a surface patch out of the straight skeleton of one cell of the slices’ overlay. In Section 5.5 we provide the running-time and space analysis of the algorithm. In Section 5.6 we present some experimental results. We note that our algorithm is general and can handle slices with multiple polygons, in any level of containment hierarchy and with any geometries. The algorithm is given in its full generality in [BGLS04].

5.2 Overview of the Algorithm

Our proposed algorithm consists of the following steps:
1. **Analyzing the contour overlay.** Compute the overlay of the two slices. For each cell in the arrangement of polygons, attach a tag that identifies whether the cell lies in the material or the nonmaterial regions of each slice. Discard all the cells that either belong to the material or to the nonmaterial regions in both slices.

2. **Surface interpolation.** Compute the straight skeletons of all the remaining cells, separately triangulate each region in the maps induced by the skeletons, and lift the triangulations up to three dimensions.

The following two sections describe the algorithm steps in detail.

### 5.3 Analyzing the Contour Overlay

We compute a representation of the arrangement of the polygons of the two slices, obtained by projecting one slice onto the other (along the $z$ direction). This can easily be done by applying a line-sweep procedure on the overlay of the two slices. As part of sweeping the plane, each cell in the arrangement is given attributes that indicate whether it lies in the material or nonmaterial regions of each of the slices.

We then discard all the cells that belong either to the material or to the nonmaterial regions of both contours. Thus, we are left with only those cells that correspond to interior of one contour and exterior of the other. Denote these as the *active* cells. Figure 5.2 shows two contours, their overlay, and the active cells of the overlay.

For the moment we will ignore the original polygon vertices and consider only the vertices of the polygon arrangement (the intersection points of the two polygons, one of each slice). By ‘contour portion’ we mean a subpolygon whose endpoints are two such vertices (intersections of the original polygons) and whose interior is free of other vertices. The endpoints of the contour portions are seen in Figure 5.2(c).
Figure 5.2: Active cells in the overlay of two slices
Theorem 5.2 Each contour portion belongs to exactly one active cell.

Proof. Consider any contour portion $AB$, where $A$ and $B$ are vertices of the arrangement of contours (intersection points of the contours of the two slices). Assume without loss of generality that $AB$ belongs to a contour of the first slice. Thus, it is shared by two cells of the arrangement, exactly one of which is material in the first slice. In the second slice, $AB$ is fully contained in either the material or nonmaterial region. In either case, by definition, $AB$ bounds exactly one active cell. □

Since we will use the boundaries of only the active cells for interpolating a surface between the two slices, we are now guaranteed that every polygon portion will be used exactly once as a boundary of that surface. Together with the original polygons, we will have a closed surface bounding a solid model.

5.4 Surface Interpolation

5.4.1 Skeletons and triangulations

At this point we have already found the boundaries of the interpolated surface. Our current goal is to construct a collection of pairwise-disjoint non-self-intersecting surface patches with known boundaries (the active cells). This is easy to achieve by forming a surface whose $xy$ projection is simple, that is, every vertical line intersects the surface in at most one point. For ease of exposition we first describe the computation of the $xy$ projection of the surface, and only then, its lifting up to three dimensions.

The $xy$ projection of the interpolated surface is simply the union of all the active cells in the arrangement of all the contours, as is shown in Figure 5.2(d). We explain in detail how to create the triangulations of these cells (which after lifting up to three
We begin by computing the straight skeletons of all the active cells. Obviously, by construction, every face in the subdivision induced by the straight skeleton of a cell contains exactly one original polygon edge, and the face is monotone with respect to that edge \cite[Lemma 3]{AAAG95}. Then, among all possible triangulations of the face, we choose a triangulation that is monotone with respect to the polygon edge (see \cite[pp. 55–58]{BKOS97}). Figure 5.3 shows two examples of an overlay of two contours (shown with regular and thick lines), the straight skeletons of the active cells (shown with dashed lines), and their respective triangulations (shown with dotted lines). We chose this triangulation because it guides an intuitive reconstruction of a surface. However, any other triangulation will do. The union of the triangulations of all the skeletal faces of the active cells (guided by the respective skeletons) is the $xy$ projection of the sought triangulated surface.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure53}
\caption{Active cells: Their straight skeletons and triangulations}
\end{figure}

\footnote{Although Lemma 3 of \cite{AAAG95} guarantees that the face is monotone with respect to its defining edge, the endpoints of the edge are not necessarily visible by all vertices of the face. (That is, both chords connecting a vertex of the face to the two edge endpoints penetrate to the outside of the face.) Thus, we either compromise the monotonicity of the triangulation (where this is unavoidable), or enforce it by introducing Steiner points along the edge. The latter operation at most doubles the number of vertices of the face, so asymptotically it does not affect the complexity of the algorithm.}
5.4.2 Lifting up

We assume without loss of generality that the lower and upper polygons are at heights 0 and 1, respectively. In order to perform our final step—that is, to lift the surface up to three dimensions—all we have to do is to assign $z$ coordinates to all the vertices of the straight skeletons. The following theorem, which refers to active cells bounded by portions of polygons of both slices, will help us in doing so. The term “offset distance” stands here for the amount a polygon needs to be offset inward in order to hit a point.

**Theorem 5.3** Let $H$ denote the vertices of the straight skeleton (of some active cell) that are offset-equidistant from the two polygons of both slices. Then the straight skeleton of the cell is the union of

1. Edge-disjoint trees whose roots are points in $H$; and

2. Skeletal edges connecting pairs of points in $H$.

This claim is shown in Figure 5.4. The portions of the two polygons which bound one cell are shown in regular and thick lines. The straight skeleton of the interior of the cell is divided into trees by vertices equidistant from the two polygons. Two nontrivial trees are highlighted. Segments connecting points in $H$, and the trivial trees (line segments) are shown with dashed lines.

**Proof.** The claim follows from the properties of the straight skeleton of a simple polygon. The skeleton itself is a tree. Removing $A$, the connected set of all edges connecting skeleton vertices that are offset-equidistant from the polygon, cuts the skeleton into edge-disjoint disconnected subtrees, and each such subtree touches $A$ at a single vertex—the subtree’s root.

In assigning $z$ coordinates to vertices we distinguish between three cases:
Figure 5.4: Trees in the skeleton, rooted at points (shown as black disks) equidistant from the two polygons of the two slices

1. Original polygon vertices. Here we naturally assign to the vertices the height of their respective polygon, that is, either 0 or 1.

2. Internal vertices of the straight skeleton. Here we have three subcases:

   (a) Vertices of $H$. We set the height of these vertices to 0.5.

   (b) Skeleton vertices that are not in $H$. According to Theorem 5.3, these are internal vertices of trees, the heights of whose roots were already set to 0.5, and whose leaves are all at height either 0 or 1. Any monotone function can be used for setting the heights of the internal vertices of the trees. To reflect the relation to the straight skeleton, we use the offset distance function (see [BDG97]) from the contour, and normalize it so that its value is 0 or 1 on the contour and 0.5 at the root of the tree. Figure 5.5 shows a close-up of the skeleton tree at the bottom-right corner of Figure 5.4. The height of the vertices $u$, $v$, and $w$ is 0 since they belong to the lower slice. The height of the root $r$ is set to 0.5, whereas the heights of the internal
Figure 5.5: Setting vertex heights according to their offsets

tree vertices $s$ and $t$ are set to $\frac{3}{5} \cdot 0.5 = \frac{1}{5}$ and $\frac{5}{9} \cdot 0.5 = \frac{5}{18}$, respectively.\textsuperscript{2}

This choice of the $z$ function fits our application due to the strong relation between the offset of a shape and its straight skeleton.

(c) The special case of an active cell bounded completely by a contour of one slice indicates the vanishing or appearing of a feature of the three-dimensional object. Assume without loss of generality that the active cell is defined by a contour of the lower slice. Then, all the leaves of the skeleton are already assigned the height 0. We set the height of the skeleton vertex (or vertices) offset-wise furthest from the contour to 1, and use, as in the previous case, the skeleton to guide the setting of the intermediate heights. We believe that setting the “vanishing height” to 1 is better than setting it to 0.5 (or to any other intermediate value in the open interval $(0, 1)$), since this gives a smooth and continuous interpolation between the two slices, morphing a feature in one slice to its absence in the other slice. The other case (active cell modelled by a contour in the upper slice) is completely

\textsuperscript{2}The values $3/9$ and $5/9$ follow from the fact that the skeleton vertices $s$ and $t$ are located 3 and 5, respectively, inward-offset units away from the polygon, while 9 units are needed to reach the root $r$. Obviously we must take care that the height of isolated branches of the tree does not exceed 1. For example, if the vertex $t$ in Figure 5.5 were twice as far as $r$ (offset-wise) from the boundary of the lower slice, which is possible since the polygons are not necessarily convex, then its height would be more than 1. In this case we apply a secondary scaling on every isolated subtree.
3. Intersection points of the pair of polygons, one of each slice. In fact, such a point is actually three points whose $xy$ projections match each other. Two of the points lie on the original polygons, and thus have their original heights (0 and 1). The third point is a skeleton point which is set to height 0.5. Note that the two points on the original polygons are not necessarily vertices of the input polygons, but only the intersection point of their $xy$ projections. Figure 5.6(a) shows the $xy$ projection of two slices, with a contour-intersection point $v$. Figure 5.6(b) shows the same scene in a perspective view. The point $v$ is actually the $xy$ projection of three points $v$, $v'$, and $v''$ (at heights 0.5, 1, and 0, respectively). The two supposed triangles in Figure 5.6(a) that share the vertex $u$ are actually quadrangles, as is seen in Figure 5.6(b). Therefore, we triangulate them by adding two more diagonals. In addition, the skeleton edge that coincides with the skeleton vertex $v$ ($uv$ in the figure) is deleted so as to form one triangle containing the edge $v'v''$ ($\Delta uv'v''$ in the figure).

After assigning the $z$ values (heights) to all the vertices of the skeletons, we lift the collection of triangulated patches up to three dimensions. The result is the desired interpolation.
The algorithm does more than described above. It operates on any kind and number of contours, and handles all branching situations and hierarchical structures. It is guaranteed to interpolate a valid surface for any possible input, and is intuitive in the sense that it tends in practice to minimize the surface area of the reconstruction. This is because it uses an offset distance function to locally decide which contour features to bind.

5.5 Complexity Analysis

We measure the complexity of the algorithm as a function of $n$, the total complexity (number of vertices) of the two polygons. We also denote by $k$ the complexity of the overlay of the two polygons. In the worst case $k$ can be as high as $\Theta(n^2)$, but in most practical cases it is $O(n)$.

Computing and analyzing the overlay of the two polygons takes $O(n \log n + k)$ time [Ba95]. This already includes the selection of the active cells of the overlay. Computing the straight skeletons of all the active cells can theoretically be done in $O(k^{17/11+\varepsilon})$ time, for every $\varepsilon > 0$ [EE99], by using a sophisticated data-structure for ray-shooting queries, or even slightly better (in nondegenerate cases and on the average) in $O(k^{3/2} \log k)$ expected time [CV02]. However, we implemented the algorithm of [FO99] whose running time is $O(k^2)$ in the worst case, and in practice, much less than that. (Our experiments suggest that this step is subquadratic in $n$.) Triangulating the monotone subcells induced by the skeletons, as well as lifting the triangulations up to three dimensions to form the interpolating surfaces, take $O(k)$ time.

To conclude, the entire algorithm runs in $O(k^2)$ time. In the worst case this is

\[^{3}\text{In fact, the precise running time is } O(k^{1+\varepsilon} + k^{8/11+\varepsilon} r^{9/11+\varepsilon}), \text{ where } r \text{ is the number of reflex polygon vertices.}\]
\(\Theta(n^4)\), but for most cases (in which the complexity of the slice overlay is linear in that of the original slices) the running time of the algorithm is theoretically \(O(n^2)\), and in practice even less than that (around \(O(n^{1.6})\) time in our experiments).

The space complexity of the algorithm is naturally \(\Theta(n + k)\), since the number of internal skeleton vertices is linear in the number of vertices of the overlay of the polygons. In the worst case this quantity is \(\Theta(n^2)\), but in practice it is proportional to \(n\).

5.6 Experimental Results

We implemented the entire algorithm in C++ on an HP Omnibook 6000 (a portable computer). The computer was equipped with a Pentium III 850 MHz processor, 128 megabytes of memory, and an ATI Rage M1 AGP Mach 64 graphics card with 32 megabytes of memory. The implementation took about two months, and the software consisted of about 8,500 noncomment lines of code. We experimented with the algorithm on several data files obtained by medical scanners, and obtained very good results in practically all cases.

Here are some specific examples of the performance of the algorithm

Figure 5.7(a) shows the overlay of two contours belonging to two successive slices (in black and grey). Since the two contours are nested, the single active cell is the ring bounded by the two contours. Its straight skeleton is shown with thick black lines. Figure 5.7(b) shows the triangulated ring.

Figure 5.8(a) shows an overlay of two complex (multiple-contour) slices (one in black and the other in grey) taken from a lungs data file. A close-up of the area in the middle of the overlay is seen in Figure 5.8(b). Figure 5.8(c) shows in thick black lines the straight skeletons of all the active cells in the arrangements of contours of
Figures 5.7(a,b) show wire-frame and shaded displays of the fully-reconstructed pair of lungs. These data contained thirty slices, thus we invoked our algorithm 29 times. Table 5.1 displays statistics of these experiments. The running times of some stages were negligible and are thus omitted in the table. The experimental results show clearly that the most time-consuming step was the computation of the straight skeleton. In our implementation it indeed required time which was asymptotically quadratic in the size of the input, while all the other steps required time linear in the input size. This is clearly demonstrated in Figure 5.10, which shows a few relations between running times and output size to complexities of the input. (The displayed functions were approximated by the curve-fitting tool of Microsoft Excel.) Overall, every layer was interpolated on average in less than one second.

Figure 5.11 shows a reconstruction of part of a human pelvis.
Figure 5.8: A complex branching example
Figure 5.9: A fully reconstructed pair of lungs
<table>
<thead>
<tr>
<th>Slice Num.</th>
<th>Contour Edges</th>
<th>Skeleton Edges</th>
<th>Surface Triangles</th>
<th>Line Sweep</th>
<th>Sym. Diff.</th>
<th>Height Setting</th>
<th>Triang. Setting</th>
<th>Total Time (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>34</td>
<td>80</td>
<td>110</td>
<td>0.05</td>
<td>0.06</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>2</td>
<td>30</td>
<td>72</td>
<td>98</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
</tr>
<tr>
<td>3</td>
<td>26</td>
<td>68</td>
<td>96</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>4</td>
<td>58</td>
<td>64</td>
<td>86</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>252</td>
<td>337</td>
<td>0.22</td>
<td>0.05</td>
<td>0.05</td>
<td>0.27</td>
<td>0.05</td>
</tr>
<tr>
<td>6</td>
<td>128</td>
<td>318</td>
<td>401</td>
<td>0.44</td>
<td>0.06</td>
<td>0.06</td>
<td>0.50</td>
<td>0.06</td>
</tr>
<tr>
<td>7</td>
<td>149</td>
<td>386</td>
<td>481</td>
<td>0.49</td>
<td>0.49</td>
<td>0.49</td>
<td>0.49</td>
<td>0.49</td>
</tr>
<tr>
<td>8</td>
<td>149</td>
<td>378</td>
<td>471</td>
<td>0.05</td>
<td>0.55</td>
<td>0.55</td>
<td>0.55</td>
<td>0.55</td>
</tr>
<tr>
<td>9</td>
<td>162</td>
<td>414</td>
<td>523</td>
<td>0.05</td>
<td>0.66</td>
<td>0.66</td>
<td>0.66</td>
<td>0.66</td>
</tr>
<tr>
<td>10</td>
<td>164</td>
<td>430</td>
<td>532</td>
<td>0.77</td>
<td>0.06</td>
<td>0.06</td>
<td>0.83</td>
<td>0.06</td>
</tr>
<tr>
<td>11</td>
<td>162</td>
<td>413</td>
<td>512</td>
<td>0.66</td>
<td>0.05</td>
<td>0.05</td>
<td>0.71</td>
<td>0.05</td>
</tr>
<tr>
<td>12</td>
<td>174</td>
<td>430</td>
<td>577</td>
<td>0.77</td>
<td>0.05</td>
<td>0.05</td>
<td>0.82</td>
<td>0.05</td>
</tr>
<tr>
<td>13</td>
<td>189</td>
<td>466</td>
<td>632</td>
<td>0.06</td>
<td>0.77</td>
<td>0.77</td>
<td>0.77</td>
<td>0.77</td>
</tr>
<tr>
<td>14</td>
<td>229</td>
<td>579</td>
<td>769</td>
<td>1.27</td>
<td>0.05</td>
<td>0.05</td>
<td>1.32</td>
<td>0.05</td>
</tr>
<tr>
<td>15</td>
<td>238</td>
<td>588</td>
<td>850</td>
<td>0.06</td>
<td>1.15</td>
<td>1.15</td>
<td>1.26</td>
<td>1.26</td>
</tr>
<tr>
<td>16</td>
<td>233</td>
<td>588</td>
<td>826</td>
<td>0.06</td>
<td>1.32</td>
<td>1.32</td>
<td>1.43</td>
<td>1.43</td>
</tr>
<tr>
<td>17</td>
<td>257</td>
<td>628</td>
<td>940</td>
<td>0.06</td>
<td>1.43</td>
<td>1.43</td>
<td>1.60</td>
<td>1.60</td>
</tr>
<tr>
<td>18</td>
<td>263</td>
<td>681</td>
<td>1,028</td>
<td>0.05</td>
<td>1.76</td>
<td>1.76</td>
<td>1.86</td>
<td>1.86</td>
</tr>
<tr>
<td>19</td>
<td>265</td>
<td>681</td>
<td>963</td>
<td>0.06</td>
<td>1.43</td>
<td>1.43</td>
<td>1.54</td>
<td>1.54</td>
</tr>
<tr>
<td>20</td>
<td>259</td>
<td>664</td>
<td>961</td>
<td>0.06</td>
<td>1.48</td>
<td>1.48</td>
<td>1.59</td>
<td>1.59</td>
</tr>
<tr>
<td>21</td>
<td>228</td>
<td>560</td>
<td>815</td>
<td>0.06</td>
<td>1.21</td>
<td>1.21</td>
<td>1.38</td>
<td>1.38</td>
</tr>
<tr>
<td>22</td>
<td>212</td>
<td>540</td>
<td>784</td>
<td>0.93</td>
<td>0.05</td>
<td>0.05</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>23</td>
<td>230</td>
<td>578</td>
<td>835</td>
<td>0.06</td>
<td>1.10</td>
<td>1.10</td>
<td>1.21</td>
<td>1.21</td>
</tr>
<tr>
<td>24</td>
<td>245</td>
<td>608</td>
<td>956</td>
<td>0.06</td>
<td>1.37</td>
<td>1.37</td>
<td>1.49</td>
<td>1.49</td>
</tr>
<tr>
<td>25</td>
<td>242</td>
<td>608</td>
<td>889</td>
<td>0.05</td>
<td>1.32</td>
<td>1.32</td>
<td>1.42</td>
<td>1.42</td>
</tr>
<tr>
<td>26</td>
<td>234</td>
<td>594</td>
<td>864</td>
<td>0.06</td>
<td>1.15</td>
<td>1.15</td>
<td>1.27</td>
<td>1.27</td>
</tr>
<tr>
<td>27</td>
<td>230</td>
<td>586</td>
<td>801</td>
<td>0.05</td>
<td>1.16</td>
<td>1.16</td>
<td>1.26</td>
<td>1.26</td>
</tr>
<tr>
<td>28</td>
<td>249</td>
<td>615</td>
<td>899</td>
<td>0.06</td>
<td>1.26</td>
<td>1.26</td>
<td>1.43</td>
<td>1.43</td>
</tr>
</tbody>
</table>

| Total     | 5,350         | 13,391         | 18,812            | 0.33       | 0.58      | 25.77         | 0.20             | 1.12                |

Table 5.1: Performance of the algorithm (empty entries are practically 0)
Figure 5.10: (a,b,c) Skeleton creation time, number of skeleton edges, and number of output triangles, respectively (in the lungs model), as functions of the number of input contour edges; (d) Number of output triangles as a function of intermediate skeleton edges.
Figure 5.11: A reconstructed pelvis
Chapter 6

Concluding Remarks

In this thesis we investigated the problem of reconstructing a polyhedral surface from two polygons lying within two parallel slices.

In the first part we explored several problems related to the existence of mating. We presented a pair of non matable polygons (of complexity smaller than that of the only example that existed up until now) with a complete proof of its non matability. We also provided, for the first time, a method for building a family of non matable pairs of polygons with unbounded complexity. We presented several sufficient conditions for the matability of two given polygons. Some conditions apply for certain classes of polygons, that is, every two polygons of the specific class are matable. Some conditions take under consideration the position of one polygon relative to the other. These conditions apply only for specific pairs of polygon.

In the second part of the thesis we allow Steiner vertices in the surface-reconstruction process. We show that given a pair of nonmatable polygons, such that one polygon has three consecutive vertices on its convex hull, it is sufficient to add only one Steiner vertex on the boundary of other polygon so as to make the polygons matable. We also proposed an efficient algorithm for solving the practical problem of polyhedral
interpolation between parallel polygonal slices.

There are several issues that we leave for future research:

1. In Chapter 3 we presented a few sufficient conditions for mating. Is there any condition (which can be checked in polynomial time) that is both sufficient and necessary? In other words, what is the computational complexity of deciding whether or not two polygons can mate? The problem of whether or not two polygons can mate in a double-cone manner (described in Chapter 2) is polynomial. However, we suspect that the absence of a double-cone mating does not guarantee the absence of any mating at all.

2. It was proven by Geiger that for any two polygons, the addition of two Steiner points suffices to guarantee mating. We showed that if one polygon contains three consecutive vertices on its convex hull, then one additional Steiner point always suffices. We also believe that if the rotation of one polygon is allowed, one Steiner point suffices. Can this be proven?

3. In Chapter 5 the use of straight skeletons may create undesired long peaks at the vicinity of sharp corners of the input polygons. To remedy this we can use, instead of the straight skeleton, a linear approximation of the medial axis, or even the chordal axis, of the cells of the symmetric difference.
Bibliography


Appendix A

Automatic Verification

In this appendix we describe a recursive algorithm that traverses possible candidate connections between two given polygons. We fully implemented this algorithm during this research. In practice, the algorithm does not traverse all possible connections between the two polygons. The algorithm constructs the connection by adding one triangle at a time. Then, if a newly-added triangle intersects with any of the existing triangles, then the algorithm back-tracks by removing this triangle and continuing to the next possible triangle. However, in the worst case, the time complexity of the algorithm is as if it checked all possible connections. In what follows we explain three functions of the algorithm. The first function, \texttt{RecursiveCheck}, is the implementation of the algorithm. The second function, \texttt{IntersectionWithNewTriangle}, and the third function, \texttt{IsUncConnectedDoubleCone}, are used for the verification of the current connection.

The function \texttt{RecursiveCheck} constructs and checks recursively the connections. The function inputs are:

1. Poly[2]: The two polygons;
2. **MatingResultList:** The current list of triangles that defines the current partial connection;

3. **Na:** The number of edges in the current connection that belong to the first polygon [0];

4. **Nb:** The number of edges in the current connection that belong to the second polygon [1];

5. **MaxA:** The complexity of the first polygon;

6. **MaxB:** The complexity of the second polygon; and

7. **TriangleToAdd:** The next triangle to be added to the connection.

The function first attempts to create triangles whose bases are edges of the first polygon. The execution of the function starts with checking if the new triangle intersects any triangle of the current partial connection; this is done by calling the function **IntersectionWithNewTriangle**. If no intersection is encountered, the triangle is added to the connection, and the function recurses. The second part of the function creates, in the same manner, triangles whose bases belong to the second polygon.

The third part of the function verifies that each edge of the two polygons is connected to some vertex of the other polygon, and checks that the connection is not a double-cone. In case we reach this part due to an intersection, the above checks are skipped, the last added triangle is deleted from the current connection, and the function back-tracks. Otherwise, the above checks are performed. Failure in either condition results in the same back-tracking.
BoolRecursiveCheck(Poly[2],MatingResultList,Na,Nb,MaxA,MaxB,TriangleToAdd) 
{
    //START
    if (IntersectionWithNewTriangle(TriangleToAdd, MatingResultList))
        return(false);
    AddTriangleToMatingResultList (TriangleToAdd, MatingResultList);
    //PART I
    if (Na<MaxA) {
        Na++
        TriangleToAdd = Set NewTriangle (Edge On Polygon A);
        If (RecursiveCheck(Poly[2],MatingResultList,Na,Nb,MaxA,MaxB,TriangleToAdd))
            return (true);
        Na--;
    }
    //PART II
    if (Nb<MaxB ) {
        Nb++;
        TriangleToAdd = Set NewTriangle (Edge On Polygon B);
        If (RecursiveCheck(Poly[2],MatingResultList,Na,Nb,MaxA,MaxB,TriangleToAdd))
            return (true);
        Nb--;
    }
    //PART III
    if (Na<MaxA || Nb<MaxB
        || IsUnConnectedDoubleCone(MatingResultList,Na+Nb)) {
        if (MatingResultList.GetTail()!=NULL){
            MatingResultList.DeleteElement(MatingResultList.GetTail());
            return(false);
        } return(true);
    }
}

The function \texttt{IntersectionWithNewTriangle} traverses all the triangles in the current connection and checks whether they intersect with the triangle to be added. The new triangle cannot intersect the previously-added triangle (the one with which it shares an edge). Also, our experience shows that in practice, if the new triangle invalidates the connection, then this is usually because it intersects with a relatively “new” triangle (close in the list). Thus, to optimize the running time, we check the list for intersections with the new triangle from tail to head.
The function `IsUnConnectedDoubleCone` checks whether the current connection is a double-cone mating. The function counts the transfers between edges of the two polygons. If the number of transfers is less than or equal to 2, then we have a closed double cone.

```c
Bool IsUnConnectedDoubleCone(MatingResultList, size) {  
    CurrentTri= MatingResultList.GetHead();  
    NextTri = CurrentTri->Next(); Jump=0;  
    while (NextTri !=NULL) {  
        if (PolygonIndexOfTriangleBaseEdge(CurrentTri)!=
            PolygonIndexOfTriangleBaseEdge(NextTri))
            jump++;  
        CurrentTri= NextTri ; NextTri = NextTri->Next(); }
    //checking last triangle with the first one on the list
    //-----------------------------------------------------------
    if (PolygonIndexOfTriangleBaseEdge(CurrentTri)!=
        PolygonIndexOfTriangleBaseEdge(MatingResultList.GetHead()))
        jump++;
    if (jump<=2) return(true);
    return(false);
}
```
Straight Skeletons

Geiger and Steiner
קימז זידוג

עבדוות רבות נוספות על למענunicipio א發布浊, שיתוף של עם פיתוי, עזרה של
הנקודות המרחקים, התוצאות מגוונותagt נלמודים, אך לא למצב של שיתוף מידה.
ות הלחימה atl בת מצבי כל הקימיים, המפרק את עלות שיתוף מידה.
הסחה לחקים ביניהם כל השימור, להכין את השיווק בנקיניים.
ולשימור משלימים, העדות והنتشرות הללו עד מצבי ביניים יוז. דוגמה זו נבנה במדע התוכנית מחשב.
בשעון זו לא מציגה התוכנה של קים ויובי עם פל도록ים ושוטפים בחירה, בנו-
עד מספר קודקדים של קים יורי מדימה בשני פל도록ים של השי
סם, א认购מצי, בטחונוראשונה, פל도록ים שביניהם משמעת של קים פל도록ים
הסירה-יווי, בעל סיבוכיות של כל דרגה.

תנאיים מספייקים לקים זידוג

בחלק מצמד פל도록ים. עביו התחברת, אם קים ליח יוזו ואלא, היא בין
longleftrightarrow המרחק והינו התוכנה המכסית קים יוזו בחרת ממולא.
עיו התחברת. 일본ית, מחזיקה נהגי צמוד טעה ועביה, המטרו פועלת אך
כ-כל חלב מצמד פל도록ים שהטייעה את קים יוזו. סימביוט בגיל זה
שבגון והתוכנה של קים יוזו עם התוכנה של קים יוזו. מכימם קים יוזו
לשם ופלוקה השיטה של המっこ פל도록ים. יוזו כלים, נוכחיים ליגודו
בי.
كفיז

מביא

ביתינית רפז של התחים מפריעים מكلفולים. אנו מんじゃないים לחשב משטח פוליה.

דרל המודעות והמודעות על כל חחק בכל שם. בהיותנו גורדים לטיפוס

פיזיוס של חקרים ביניהם מחזיקה שלגיות האתגרות. הביעה עליהبعיך

בתוחנה המודעות הרפואית ומרוחקת מורכבות מנורא. המידה. המתקבלי ממקויה

המודיעה הרפואית המسكانית פנינו אפילו. עונר אינטפורסילמה על שעה שלחר להזין

אבלים או גורסים או אף שטח טופוגרפיים. בכל אלכונרואים השחורים הידיעות קומ

והנה להנהיג המודע עבורה קמקולות ומקימ הל潛ונים חتكاملים מחזיקה מחזיקה תסאר

כל ממלוכ מפריד ביניהם של "חורר" לאול של "חוחר" "אני חוחר" המטרה. אם, היא

השישה משלום המחבר ביניהם חила המצלימים ויצר נפח שניחנה בברחים

המפוודים חזרה עד החכמה המสะดวก. עבורה והגור גורגל של שחרור משה.

עת עדבר שייתחפזים מחזיקה שלחד מאה מחזיקה חcreateCommand

התקדר האישה של המחבר ריכס למיסים של שחרור הנקרא חוח. בחרניא

ווג מצלילים בשני מישורים מكلفולים נותרו. יוזג כמות המצללים שיאיה חほど

את עצומה. המזרחי על.connkドיפוס של שני המצללים בלבם. מתמזג בחלקה

הזווית המצללים קסום של שום בינו הם יזי. יוזג, מздравה משפה של

זווית המצללים קסום-יוזג על-ניקופים גולת קרצונה. חוויתו תנו מצלולים,

רסחיים המצללים להודיע לו חים שלנו היא קשה. מחקרי הו אנימיסים ממסס תנים

מספיקים לכולים יוזג בין מצלולים.

התקדר האישה של המחבר ריכס למיסים של שחרור הנקרא חוח. בחרניא

ענ. מחקרי הו אנימיסים ממסס תנים המצללים שלנו היא קשה. יוזג, מздравה

connkドיפוס-יוזג על-ניקופים גולת קרצונה. חוויתו תנו מצלולים

ביית הסמוכיות האמא קים שלנו היא קשה. מחקרי הו אנימיסים ממסס תנים

מספיקים לכולים יוזג בין מצלולים.
ראות משותרות 5.9
אפ inaugur תשלד, מספר ההזעה של תשלד, מספר המשר 5.10
לעשים שני פעמים בתחום 5.11
שוחה של אנ
35 הקודקוד הממעגר \((q_1, q_2)\) למază במנזר (למה 4.1)
36 הקודקוד הממעגר \((q_1, q_2)\) למزة ביצת (למה 4.2)
36 הקודקוד הממעגר \((q_1, q_2)\) למزة ביצת (למה 4.3)
37 התצל \(q_2 \rightarrow q_1\) מודגשים (למה 4.4)
38 ... \(Q\) البعז יבזנבחר לקודקוד הקרבונ של הטוליגן
39 התצל \(q_0 \rightarrow q_2\)
40 מותרים \(a\)
41 התצל \(q_2 \rightarrow \triangle p_{p_0 q_0}\)
42 התצל \(q_2 \rightarrow \triangle p_{p_0 q_1}\)
43 התצל \(q_2 \rightarrow \triangle p_{p_0 q_1}\)
44 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
45 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
46 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
47 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
48 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
49 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
50 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
51 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
52 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
53 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
54 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
55 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
56 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
57 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
58 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
59 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
60 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
61 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
62 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
63 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
64 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
65 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
66 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
67 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
68 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
69 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
70 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
71 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
72 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)
73 \(\triangle p_{p_0 q_0}\) \(\triangle p_{p_0 q_1}\) \(\triangle p_{p_0 q_1}\)

ן
רשימה אפורים

10 .................................................. 2.1
11 .................................................. 2.2
12 .................................................. 2.3
13 .................................................. 2.4
17 .................................................. 3.1
19 .................................................. 3.2
20 .................................................. 3.3
21 .................................................. 3.4
21 .................................................. 3.5

3.6
3.7
3.8
3.9
3.10
4.1
59  מובא 5.1
61  סקירה של האלגוריתם 5.2
62  נימוקים מקדמת המצליעם 5.3
64  שחזור המשטח 5.4
64  חישוב שלד ומיצאת שלוש 5.4.1
66  המטרת השילוח 5.4.2
66  נימוק סיבוכיות 5.5
71  תוצאות ניסיונית 5.6

6  סיכום

81  רשימה מקוורת
חותן עניינים

תקציר

 mobs

 buurt מצוקים

 תצוגת מרצות

 תואם מפשים לקוור דרור

 מתכון של מקטילים בוליב שא

 מפייה דמייה-רכה

 מפייה בוליב-תחולק

 גנוט של מפלטלים בוליב גייל

 דרגמה למגנובלים חסרי דירה

 תיון כלל

 תקנות

 אמ יהודה פאר אזרם

 משפחות ממפלטלים חסרי יוזג

 Straight Skeletons

 אלגוריהות שתרור בערות

 או
וףחק נעש הבניה פורפסור על ברקת הפקולתה لمدة المتحשב

ברגוני להוות ליגל ברקת על שבחר בתי להוות סטודנטית بغיהות. אני מודע
על הווירטואליות לכל משך מחקר ברמה גבוהה, על העברה, החינוך והמטופלים.

המיומנויות והורישיות

אני מודע לפיכך על ההמימה הכספית הגיזה בתרומתו.
ɥὶὸὶ ṣ lofty
חיבר על מחק
לשםupilウィシェルקיעשל הדירישית לכותבת החרוא
מגייסר למדעיכם במדעטיית שימישית
איה סטיוין

הוגש לטסהה הסכויות - מכון טכנולוגי לישראל
חג ש"ה 177חף אוקטובר 2004
звонיות של פוליגון

איך שטני

ר