Polygon Reconstruction from Line Cross-Sections

Avishay Sidlesky
Polygon Reconstruction from Line Cross-Sections

Research Thesis

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

Avishay Sidlesky

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

Nisan 5766 Haifa April 2006
The Research Thesis was done under the supervision of Prof. Gill Barequet and Prof. Craig Gotsman in the Faculty of Computer Science.

I would like to thank Craig and Gill for the privilege of having such a fascinating research topic, for their devoted guidance, and for everything they taught me throughout the research. I would also like to thank them for their courage in supervising me as an external student.
## Contents

Abstract

1 Introduction

2 Polygon Reconstruction

2.1 Problem Definition

2.1.1 The sampling condition

2.2 The Reconstruction Algorithm

2.2.1 Definitions

2.2.2 Data-structure initialization

2.2.3 Bridge-cell configurations

2.2.4 Polygon reconstructions

2.3 Algorithmic Notes

2.3.1 Trimming corners

2.3.2 Ambiguous local geometry

2.4 Implementation Notes

2.4.1 Data-structure implementation
List of Figures

2.1 Real-life instance of the polygon-reconstruction problem . . . . . . . . 8
2.2 Close-up on a corner of a polygon . . . . . . . . . . . . . . . . . . . . . . 9
2.3 Cell and signature . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
2.4 Cell classification . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
2.5 All configurations of a bridge cell having three signature chains . . . 12
2.6 Cell reconstructions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
2.7 Cases of a compulsory vertex . . . . . . . . . . . . . . . . . . . . . . . . . 15
2.8 Eliminating bridge-cell configurations . . . . . . . . . . . . . . . . . . . 15
2.9 Invalid combinations of two adjacent cell configurations . . . . . . . 16
2.10 A possible reconstruction of two adjacent cells . . . . . . . . . . . . . . 17
2.11 Trimming corners . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
2.12 Inconclusive local geometry . . . . . . . . . . . . . . . . . . . . . . . . . . 19
3.1 A specific example (lower bound) for the complexity term . . . . . . . 25
4.1 Input . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27
4.2 Highlighted regular cells and bridge cells . . . . . . . . . . . . . . . . . . 28
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3</td>
<td>Bridge-cell configurations and compulsory vertices</td>
<td>29</td>
</tr>
<tr>
<td>4.4</td>
<td>Four non-self-intersecting reconstructions</td>
<td>30</td>
</tr>
<tr>
<td>4.5</td>
<td>Eight reconstructions with trimmed corners</td>
<td>31</td>
</tr>
<tr>
<td>4.6</td>
<td>Four test cases</td>
<td>32</td>
</tr>
<tr>
<td>4.7</td>
<td>Illustration of the algorithm of Coll and Sellarès</td>
<td>35</td>
</tr>
<tr>
<td>4.8</td>
<td>The reconstruction of Coll and Sellarès vs. ours</td>
<td>35</td>
</tr>
</tbody>
</table>
Abstract

The field of geometric probing is defined by Steven S. Skiena as determining a geometric structure or some aspect of that structure from the results of a mathematical or physical measuring device, a probe.

We study the following geometric probing problem: Reconstruct a planar polygon from its intersections with a collection of arbitrarily-oriented “cutting” lines. This is the two-dimensional version of the problem of reconstructing a three-dimensional object from nonparallel planar cross sections which arises, e.g., in medical applications, such as hand-held ultrasound devices.

In this thesis we propose an algorithm which enumerates all possible reconstructions that are consistent with the input (cutting lines and intersection segments), and comply with a natural sampling condition. The sampling condition prevents the case of infinitely-many solutions, differing slightly from each other. We discuss the issue of multiple reconstructions in detail, and analyze the run time and space complexity of the algorithm.

The algorithm was implemented as a program in C++. The program is executed on user-defined inputs, and displays all valid reconstructions on screen.

We provide some experimental results and compare our algorithm to previous work by Coll and Sellarès. In addition, we describe a degenerated version of our algorithm that outputs the same (single) reconstruction as theirs.
Chapter 1

Introduction

In the field of medical imaging, numerous applications can benefit from a three-dimensional model of human organs. Common imaging techniques, such as MRI and CT, output a dense set of parallel slices of the region of interest and as a consequence, practically all prior art addresses the problem of reconstructing a three-dimensional triangular mesh from parallel slices. Some algorithms for three-dimensional mesh reconstruction use a volumetric approach. These algorithms essentially produce small triangles whose size is similar to that of the input voxels. Other algorithms are geometric in nature and assume a preprocessing edge-detection phase that identifies contours in each slice. These contours represent the boundary between “material” and “nonmaterial” regions. Each slice may contain several contours and contour hierarchies. Early works dealt with the case of slices having a single contour in each slice, while more advanced algorithms cover the case of multiple contours in each slice. Barequet and Sharir [BS96] give an extensive overview of work done prior to theirs. An additional survey is given by Müller and Klingert in [MK93]. Other algorithms are suggested by Bajaj et al. [BCL96] and by Barequet et al. [BGLS04].

In recent years, sensors capable of accurately measuring position and orientation
(referred to as P&O) have been developed. These sensors, when mounted on hand-
held devices, such as ultrasound transducers, provide valuable P&O information of the
cross-section planes, thus expanding the scope of previous work to the reconstruction
of a three-dimensional triangular mesh from arbitrary, nonparallel, slices. It is noted
that volumetric methods cannot be used “as-is” in the case of arbitrary cross-sections
since small equally-sized voxels cause an uneven sampling of the space. These methods
have, therefore, been extended to the concept of ‘compounding’ the cross-sections in
the volumetric representation (see, e.g., Rohling et al. [RGB97]). To the best of our
knowledge, no work that is based on the geometric approach has been done in this
direction to date.

An interesting, yet unexplored, problem in itself is the two-dimensional version of
the problem, namely, two-dimensional polygon reconstruction from line cross-sections,
which is defined as follows: Given a set of cutting lines and the intersection-segments
of the interior of an unknown polygon (or possibly several distinct or nested poly-
gons) with these lines, find the best possible polygon(s) fitting the cross-sections. A
sampling condition, that guarantees sufficient sampling of the polygon edges must be
formulated, otherwise an infinite number of solutions may exist. In addition, a merit
function may be defined to allow comparison between solutions, if several exist, to
choose the “best” one.

To the best of our knowledge, there exists only a single publication by Coll and
Sellarès [CS01] on a problem that closely resembles ours. In this paper, the authors
assume that the input crossing lines are sufficiently “dense” and uniformly distributed
over the unknown shape. They introduce an algorithm that processes the input lines
sequentially, and is incremental in the sense that for each new line it updates the
reconstruction in expected time which is logarithmic in the number of cutting lines.
At each stage, their algorithm outputs a single triangulation from which a polygonal
reconstruction of the unknown shape could be extracted. In contrast, our aim is to
treat also cases in which a small number of cross-sections are given, and enumerate all possible solutions, ordered according to some measure of quality.

Coll and Sellarès also give an overview of the related field of curve reconstruction. The even wider field of geometric probing is defined by Skiena [Sk97] as determining a geometric structure or some aspect of that structure from the results of a mathematical or physical measuring device, a probe. The key issue in geometric probing is the interaction, where the nth probe depends upon the outcome of the previous probes. Skiena also mentions several types of probes, among which are I. Point: An oracle that tests whether a given point is within an object or not; II. Finger: An oracle that returns the first point of intersection between a directed line and an object; III. Hyperplane: An oracle that measures the first time at which a hyperplane moving parallel to itself intersects an object; IV. X-ray: An oracle that measures the length of the intersection between a line and an object; V. Silhouette: An oracle that returns a (d – 1)-dimensional projection (in a given direction) of a d-dimensional object; VI. Halfspace: An oracle that measures the area or volume of the intersection between a halfspace and an object, and VII. Cut-set probe: An oracle that for a specified graph and partition of the vertices returns the size of the cut-set determined by the partition.

We may consider the line cross-sections presented here as a merge between several types of probes (e.g., a merge between advanced versions of the finger probe and the X-ray probe). Still, the main issues in geometric problems are Determination and Verification, where Determination is defined as the algorithmic problem of computing how many probes of a particular model are necessary to completely determine or reconstruct an object drawn from a particular class of objects, and Verification is defined as the algorithmic problem, given a supposed description of an object, of computing how many probes of a particular model are necessary to test if the description is valid.
Yet another related field is the combinatorial descriptions of a simple polygon and the relationships among them. Such well-known structures are internal and external visibility graphs, the set of reflex vertices, the convex hull, and the order type of the vertex set. It is known, for example, that the order type of the vertex set includes more information than the three other structures combined. Everett, Hurtado, and Noy [EHN99] present the *stabbing information* of polygons. While there are several interpretations of this term, they referred to the edges intersected by lines passing through every pair of vertices of the polygon. Everett et al. show the relationships between the stabbing information and the previously known combinatorial structures.
Chapter 2

Polygon Reconstruction

2.1 Problem Definition

The problem of two-dimensional polygon reconstruction from line cross-sections is defined as follows:

Given a set $\mathcal{L}$ of cutting lines and the intersection-segments, $\mathcal{S}$, of an unknown polygon (or possibly several disjoint or nested polygons) $\mathcal{P}$ with $\mathcal{L}$, find all reconstructions $\mathcal{R}$ that are consistent with the input, i.e., $\mathcal{R} \cap \mathcal{L} = \mathcal{P} \cap \mathcal{L} = \mathcal{S}$.

Figure 2.1 shows an instance of the problem.
We define the following terms:

- **Cutting line**: A line passing through the unknown polygon(s), partitioning the plane into two halfplanes.

- **Intersection segment**: The intersection of a cutting line and the unknown polygon(s) \( P \). There may be several disjoint intersection segments along a single cutting line. The collection of all intersection segments is called the *polygon signature*.

### 2.1.1 The sampling condition

In order to limit the number of possible solutions to the polygon-reconstruction problem, we impose a *sampling condition*. This condition requires that each edge of the reconstructed polygon(s) \( R \) be intersected by at least two cutting lines in distinct locations. The sampling condition does not guarantee a unique solution, but it prevents the creation of infinitely-many solutions, differing slightly from each other.

For instance, if some edges of a reconstructed polygon are not intersected by any cutting line, that region of the polygon can be arbitrarily reconstructed. If an
edge is intersected by a single cutting line, the intersection point becomes a pivot around which the reconstructed edge may revolve, generating an infinite number of reconstructions. See Figure 2.2.

(a) An edge that is not intersected at all  (b) An edge intersected by a single cutting line

Figure 2.2: Close-up on a corner of a polygon

**Theorem 2.1** The sampling condition guarantees a finite number of reconstructions.

**Proof.** The condition implies that the number of possible support lines for the sides of the polygon is finite. Hence, the number of possible vertices of the polygon, which lie at the intersections of the support lines, is finite as well. \( \square \)

### 2.2 The Reconstruction Algorithm

#### 2.2.1 Definitions

Before describing our algorithm, we define a few additional terms:

- **Cell**: A convex region in the plane inscribed by portions of some cutting lines, namely, a cell in the arrangement of the cutting lines. We restrict ourselves to finite cells by adding a bounding rectangle.
• **Signature segment**: A portion of an intersection segment that lies on an edge of a cell. Note that an edge of a cell may contain several (disjoint) signature segments or none at all.

• **Signature chain**: The concatenation of several signature segments joined at their endpoints and occupying multiple cell edges.

(a) A single cell having three edges  
(b) Five signature segments in two signature chains

Figure 2.3: Cell and signature

• **Boundary cell**: A cell that contains at least one signature chain on its boundary. A cell whose entire boundary is one signature chain must lie in the interior of the polygon and is thus an *interior cell*. If the boundary of the cell contains no signatures, then the cell must lie entirely outside the sought polygon and is therefore called an *exterior cell*.

• **Regular cell**: A boundary cell having a single signature chain.

• **Bridge cell**: A boundary cell with more than one signature chain. It is called this because it has the potential of connecting between two or more disconnected components of the reconstructed polygon.

(a) Input (the original polygon is shown by dotted lines); (b) Cell classification: The bridge cell in this example has two signature chains (cells are offset for clarity).
• **Cell configuration:** A matching of all endpoints of signature chains that can be realized by noncrossing chords. A regular cell has a single configuration, while a bridge cell has several configurations. The cell configuration determines the local topology of the reconstruction. Figure 2.5 illustrates all possible configurations of a bridge cell having three signature chains.

• **Cell reconstruction:** A pairwise connection of all the endpoints of a cell’s signature chains, for a specific cell configuration. Each connector is either a straight line segment, which eventually becomes a portion of an edge of the reconstructed polygon, or two consecutive line segments sharing an intra-cell vertex, which corresponds to a corner of the reconstructed polygon. The same cell configuration can be realized in more than one cell reconstruction.

• **Polygon reconstruction:** A reconstruction of all boundary cells for a specific combination of cell configurations.

Our algorithm operates only on the boundary cells. It reconstructs the polygon by creating portions of the polygon edges within each such cell. We use inter-cell relations to find all polygon reconstructions consistent with the input.
The algorithm consists of the following steps:

1. Data structure initialization;
2. Reducing the number of bridge-cell configurations; and
3. Identifying polygon reconstructions.

### 2.2.2 Data-structure initialization

The data structure is initialized in several steps (depicted in Figure 2.4):

- Initialize the cells data extracted from the set of cutting lines only (ignoring the signature segments). Each cell structure contains a list of edges and adjacency relations with its neighboring cells, maintained by a data structure similar to a DCEL.
• Add the signature segments to the data structure by subdividing the intersection segments on each cutting line at the intersection points of the cutting lines, and adding the signature segments to the relevant edge in each cell. Note that each such portion of an intersection segment belongs to exactly two cells (sharing a common edge). Every pair of neighboring cells that do not have any signature segment along their common edge are detached in the DCEL data structure and are not considered neighbors any more.

• Concatenate the signature segments in the cells so as to create chains of signatures. Since the signature chains are eventually connected at their endpoints, we no longer relate to the signature segments.

• Finally, classify the cells according to the number of signature chains on their boundaries. A cell whose entire boundary is a single closed signature lies en-
tirely inside the polygon and is thus classified as an interior cell. A cell whose boundary contains no signature at all lies outside the polygon and is thus classified as an exterior cell. Both interior and exterior cells are ignored by the algorithm. All remaining cells are boundary cells. Regular cells are cells having a single signature chain, and cells with multiple signature chains are classified as bridge cells.

2.2.3 Bridge-cell configurations

We initialize the set of configurations for each bridge cell to a fixed set. These configurations depend only on the number of signature chains in the bridge cell (see Figure 2.5). Each configuration specifies a different grouping of signature chains, where each subgroup may be composed of a single signature chain as well.

Next, we apply geometric constraints to reduce the number of bridge-cell configurations before the final step of the algorithm. While reconstructing a cell, i.e., connecting signature chains according to their partitioning, the connectors can be either a straight line segment, which is a portion of an edge of the reconstructed polygon, or two line segments sharing an intra-cell vertex, which yield a corner in the reconstructed polygon. We distinguish between these two types of connectors to reduce the number of bridge-cell configurations. This is done in two phases described below.

- First, tag the connectors in each cell’s configuration as having a compulsory vertex if either
  a. A single signature segment is connected to itself at its end; or
  b. Two collinear signature segments are connected.

See Figure 2.7(a,b), respectively.

14
• Second, to reduce the number of possible bridge-cell configurations, check all the cells having a single configuration, and delete the adjacent bridge-cell configurations if both connectors at the same signature were tagged as having a compulsory vertex (which is a contradiction to the sampling condition). Note that during this phase, as in Figure 2.8, some bridge cells may remain with only one possible configuration. Such bridge cells are simply queued and traversed as any other regular cell.

2.2.4 Polygon reconstructions

At this point we are left with a number of polygon reconstructions that is equal to the product of the number of remaining configurations of all the cells. Each potential
reconstruction that may be realized as one or more polygons stems from a distinct combination of cell configurations.

By traversing all the cells, we attempt to connect the endpoints of signature chains according to the cell configuration. If a connector was previously tagged as having a compulsory vertex, and if either

1. one of the adjacent cell connectors of the same signature endpoint was also tagged as such; or

2. an intra-cell vertex (created by intersecting the straight line connectors from the adjacent cells) is found outside the cell,

then we conclude that the entire polygon reconstruction is invalid (see Figure 2.9). If all the cells have been reconstructed correctly, this combination yields a valid solution. See Figure 2.10 for a valid local reconstruction of two adjacent cells.

(a) Two compulsory vertices for the same signature segment in adjacent cells  
(b) Reconstructed edges intersect outside the cell

Figure 2.9: Invalid combinations of two adjacent cell configurations
If one wishes to reconstruct only non-self-intersecting polygons, we check for intersections of the connectors in every cell in which at least one connector was realized with an intra-cell vertex.

2.3 Algorithmic Notes

We now discuss the issue of multiple reconstructions. The obvious reason for this phenomenon is having one or more cells (bridge cells) with multiple cell configurations and, therefore, a potentially large number of combinations of cell configurations. The worst-case number of reconstructions is analyzed in Chapter 3.

In fact, there are two other causes of multiple reconstructions that are worth mentioning. Both may yield an exponential number of reconstructions for a given cell-configuration combination if they are not treated carefully.

2.3.1 Trimming corners

It is easy to see in Figure 2.11 that all noncompulsory intra-cell vertices may be replaced by edges that connect the endpoints of the signature chains and still obtain

Figure 2.10: A possible reconstruction of two adjacent cells
reconstructions that are consistent with the input and comply with the sampling condition. If these corners are treated separately, we immediately get an exponential number of reconstructions. Therefore, our implementation optionally restricts the output to trim all corners, thus, yielding a single reconstruction per cell-configuration combination.

2.3.2 Ambiguous local geometry

In some cases, a portion of the polygon can be reconstructed with noncompulsory intra-cell vertices in adjacent cells (see Figure 2.12). In such cases, the local geometry is inconclusive and the polygon(s) can be reconstructed with a single intra-cell vertex in one of the cells or none at all. As in the case of trimming corners, this can result in an exponential number of reconstructions. Our implementation avoids this ambiguity by reconstructing the signature chains of both cells without an intra-cell vertex, effectively trimming these corners in both cells. Once again, a single reconstruction is produced for a given cell-configuration combination.
2.4 Implementation Notes

The algorithm was implemented in MEscher2D, a program written in C++, Visual Studio .Net, using the VTK library [MSL] release 4.2. The input to the algorithm is created by defining input polygon(s) and a set of cutting lines, thus, creating the intersection segments. The program then executes the algorithm and displays each reconstruction that is consistent with the input. Note that if the input does not satisfy the sampling condition, the algorithm does not reproduce the input polygon as one of the solutions.

2.4.1 Data-structure implementation

As stated above, the data structure of the cells is basically a DCEL structure. While it was possible to make use of existing implementations, such as the DCEL of the CGAL library [CGAL], we saw little point in doing so, since our implementation added a considerable amount of data on top of it. As an example, we give a partial list of additional attributes of each cell:

- Cell classification (interior, exterior, or boundary cell);
• A list of signature segments on each edge, with arclength measures of their endpoints;

• A list of signature chains, composed of the signature segments at their ends;

• Adjacency lists with the neighboring cells, including indications of adjacent signature segments and signature chains; and

• Remaining configurations of the cell.

During the reconstruction of a single combination of configurations of cells, we keep track of indications of compulsory and intra-cell vertices, which cells have already been visited, which signature chains have already been realized, etc. The extra attributes of the DCEL structure stated above simplify the access to the data structure and allow the implementation to satisfy the run time complexity stated in section 3.

2.4.2 Geometric precision issues

The algorithm was implemented with no intent to support noise in the input. However, there are certain thresholds in the implementation that prevent floating-point errors, such as for inspecting collinearity of edges and for querying intersection points to fall inside/outside the cells. These thresholds may be used to accommodate for small amount of noise in the input. Still, since the current implementation forces the edges to pass through the endpoints of the intersection-segments $S$, some long straight edges may appear fragmented in the polygonal output, so additional noise consideration should exist in the generation of the output as well (yet unimplemented).
Chapter 3

Complexity Analysis

We now provide bounds on the run time complexity of our algorithm. We first examine the space complexity of the data structure described in Section 2.2.2, and the time required to initialize it. We denote by \( \ell \) the cardinality of the set of cutting lines \( \mathcal{L} \), and by \( s \) the number of intersection segments in \( \mathcal{S} \). Building the DCEL structure of the cells in the arrangement of \( \mathcal{L} \) takes \( \Theta(\ell^2) \) time in the worst case. While generating the signature segments, each intersection segment is split at the boundaries of the cells. Due to the zone theorem (of a line in an arrangement of lines), all intersection segments that lie on a single cutting line occupy \( O(\ell) \) cells, thus \( O(\ell) \) new signature segments are introduced with each cutting line. Assuming that the intersection segments are already ordered along the cutting lines (as part of the input), it takes \( O(\ell) \) time to introduce the new endpoints of signature segments along each cutting line. We eventually have \( O(\ell^2 + s) \) endpoints of signature segments, each of which is introduced in linear time into the data structure, thus, the time required to initialize the data structure and its total space complexity is \( O(\ell^2 + s) \) as well.

To analyze the complexity of the rest of the algorithm, we need the following theorem:
Theorem 3.1  The number of configurations of a bridge with \( n \) signature chains is \( C_n \), the \( n \)th Catalan number.

Proof. One of the problems known to have \( C_n \) complexity, presented by Richard P. Stanley [St99], is stated as follows:

Partitions \( \{B_1, \ldots, B_k\} \) of \([n]\) such that if the numbers 1, 2, \ldots, \( n \) are arranged in order around a circle, then the convex hulls of the blocks \( B_1, \ldots, B_k \) are pairwise disjoint.

Our problem closely matches the above, once numbering the signature chains sequentially from 1 to \( n \) along the boundary of the cell.

For the complexity analysis, we may use the well known approximation of \( C_n \) (for \( n \gg 1 \))

\[
C_n \sim \frac{4^n}{\sqrt{\pi n^{3/2}}}.
\]

Note that this asymptotic approximation is not tight for small values of \( n \), but we can nevertheless use it since it is always an upper bound on \( C_n \).

The time complexity of the entire algorithm is dominated by the procedure described in Section 2.2.4, in which all combinations of cell configurations are traversed in order to find those reconstructions that are consistent with the input.

Let \( k \) be the number of boundary cells in the arrangement of cutting lines, and let \( m_i \) be the number of signature chains along the boundary of the \( i \)th cell, where \( \sum_{i=1}^{k} m_i = M \). We may consider \( M \) a “budget” of signature chains that is distributed among the boundary cells. Notice that \( M = 2s \) since each endpoint of an intersection segment is considered as two signature chain endpoints (see Figure 2.4(b)). The complexity, i.e., the number of configurations of a single cell, is, therefore, \( O(\frac{4^{m_i}}{\sqrt{\pi m_i^{3/2}}} \).

22
(We leave the constant term $\sqrt{\pi}$ in the $O(\cdot)$ notation since it becomes significant when considering all the cells.)

Multiplying the above expression for all $k$ cells gives the following upper bound on the number of reconstructions that are consistent with the input:

$$O\left(\frac{4^M}{\prod_{i=1}^k (\sqrt{\pi}m_i^{3/2})}\right).$$

(3.1)

To conclude this part of the analysis, we observe the values of $k$ and $m_i$’s that maximize the above term.

**Theorem 3.2** *The maximal value of the term (3.1) is obtained by minimizing the number of cells and by partitioning $M$ such that all but one of the boundary cells have a single signature chain.*

**Proof.** Since the numerator in (3.1) does not depend on $k$ and on the distribution of $m_i$’s, we should minimize the denominator of this term, $\pi^{k/2} \prod_{i=1}^k m_i^{3/2}$. This can be achieved in two manners:

Firstly, we can see that only when $k$ is large we get an exponential term whose power depends on $M$. The minimal denominator is, therefore, obtained with a low (constant) number of cells $k$. We justify that more formally below.

Secondly, we determine how $M$ should be partitioned to $m_i$’s such that the term $\prod m_i^{3/2}$ is minimized. We use Lagrange Multipliers to find the extremum of this function, with the restriction that $\sum m_i = M$.

$$f : \prod_{i=1}^k m_i^{3/2}; \quad g : \sum_{i=1}^k m_i - M = 0$$

Deriving $f$ and $g$ with respect to $m_i$ gives

$$\frac{\partial f}{\partial m_i} = 3/2 m_i^{1/2} \prod_{j=1, j\neq i}^k m_j^{3/2}; \quad \frac{\partial g}{\partial m_i} = 1$$

(3.2)

The extremum is achieved for

$$\frac{\partial f}{\partial m_i} + \lambda \frac{\partial g}{\partial m_i} = 0; \quad \forall i$$

(3.3)
Substituting Equation (3.2) into Equation (3.3) gives
\[ m_i^{1/2} \prod_{j=1; j \neq i}^{k} m_j^{3/2} = -\frac{2}{3} \lambda \quad \forall i. \]
Since the equations are symmetric with respect to \( i \) and since \( \lambda \neq 0 \), dividing any two equations from the above set provides the trivial solution in which all \( m_i \)'s are equal, that is,
\[ m_i = M/k \quad \forall i. \]
A simple inspection reveals that the extremum found is actually the maximum of \( f \). Since there is only one extremum within the range, this indicates that the minima must be on its boundary. Each \( m_i \) ranges between 1 and \( M - k + 1 \). If we set \( m_k \) to be 1, the same analysis conducted above still applies for all remaining elements \( m_1, \ldots, m_{k-1} \), where the minima must lie on their boundary.

We conclude that not only the solution is found somewhere along the boundary of \( f \), but also at the endpoints of the ranges of \( m_i \)'s, providing the highest complexity of the algorithm in the uneven partition
\[ m_i = 1 \quad (1 \leq i < k) \quad , \quad m_k = M - k + 1. \] (3.4)

We can now justify the fact that \( f(k) = \pi^{k/2} \prod_{i=1}^{k} m_i^{3/2} \) is minimized when \( k = 1 \). Its extremum is obtained when \( k = M + 1 - 3/\ln \pi \). This is a maximum point, in which \( f \) is approximately \( \pi^{M/2} \). Let us check the endpoints of the range of \( k \): \( f(M) \) equals \( \pi^{M/2} \), while \( f(1) \) equals \( \sqrt{\pi} M^{3/2} \). For sufficiently-large values of \( M \) (namely, for \( \log(M)/(M - 1) < (\log \pi)/3 \), i.e., \( M > 5 \)), we have \( f(1) < f(M) \), so \( k = 1 \) minimizes \( f \).

Substituting Equation (3.4) into Equation (3.1), we get a bound that does not depend on any specific \( m_i \):
\[ O\left( \frac{4^M}{M^{3/2}} \right). \] (3.5)
To conclude the complexity analysis, we account for the time it takes to generate each of the reconstructions, which is $O(M)$. This is because a reconstruction has to connect all signature chains’ endpoints. The time complexity of the algorithm is, therefore,

$$O\left(\frac{4^M}{\sqrt{M}}\right).$$

If we consider reconstructions that are non-self-intersecting, we spend $O(m_i \log m_i)$ additional time while reconstructing each cell. Thus, the most complex cell, that has $O(M)$ signature chains, adds $O(M \log M)$ to the bound in (3.5). The upper bound on the total time complexity of the entire algorithm, for non-self-intersecting reconstructions, then becomes

$$O\left(\frac{4^M \log M}{\sqrt{M}}\right).$$

Following Theorem 3.2, we provide a specific example which establishes a lower bound on the term (3.1).

![Figure 3.1: A specific example (lower bound) for the complexity term](image)

In this example, the original polygon that has been cut has an arbitrary number of spikes pointing upwards (three appear in the figure), and the number of cells, $k$, is three.

The number of edges, $N$, of the original polygon is

$$N = 4n_{spikes}.$$
The numbers of signature chains (in this case, signature segments) in each cell are

<table>
<thead>
<tr>
<th>Cell</th>
<th>$m_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>I,II</td>
<td>$2n_{\text{spikes}} = \frac{N}{2}$</td>
</tr>
<tr>
<td>II</td>
<td>$4n_{\text{spikes}} = N$</td>
</tr>
</tbody>
</table>

and the total number of signature segments is

$$M = 2N.$$ 

Therefore, a lower bound on the term (3.1) given by this example is asymptotically

$$\Omega\left( \frac{4^{2N}}{\pi^{3/2} \prod_{i=1}^{3/2} m_i^{3/2}} \right) = \Omega\left( \frac{2^{4N}}{\pi^{3/2} (((\frac{N}{2})^2 N)^{3/2}} \right) = \Omega\left( \frac{2^{4N+3}}{\pi^{3/2} N^{9/2}} \right) = \Omega\left( \frac{16^N}{N^{9/2}} \right),$$

which is, as a function of the number of signature chains (compare to (3.5)),

$$\Omega\left( \frac{4^M}{M^{9/2}} \right).$$
Chapter 4

Experimental Results

4.1 A Walk-Through Example

We now give a detailed walk-through example of the algorithm. Figure 4.1 shows the input confined to a bounding box. The input consists of $\ell = 8$ cutting lines with $s = 14$ intersection segments. During the initialization phase of the algorithm, we create a DCEL structure with 25 cells, of which 19 are boundary cells. The intersection segments are split at the intersection of the cutting lines and combined along the cells' boundaries to create signature chains, where $\sum m_i = 2s = 28$. The boundary cells are further classified as 12 regular cells and 7 bridge cells. The regular cells and
bridge cells are highlighted in Figure 4.2. According to Theorem 3.1, the number of configurations of a bridge cell with $m_i$ signature chains is $C_{m_i}$. The table below gives the upper bound on the number of polygon reconstructions in our example, as a product of the number of configurations of all cells.

![Figure 4.2: Highlighted regular cells and bridge cells](image)

<table>
<thead>
<tr>
<th>Cell Type</th>
<th>$m_i$</th>
<th>$C_{m_i}$</th>
<th>Multiplicity</th>
<th>Number of combinations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>1</td>
<td>1</td>
<td>12</td>
<td>$1^{12} = 1$</td>
</tr>
<tr>
<td>Bridge II</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>$2^5 = 32$</td>
</tr>
<tr>
<td>Bridge III</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>$5^2 = 25$</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td><strong>19</strong></td>
<td><strong>800</strong></td>
</tr>
</tbody>
</table>

At this point, the upper bound on the number of reconstructions equals 800. Next, the algorithm assigns compulsory intra-cell vertices for each cell configuration as depicted in Figure 4.3, and eliminates the configurations that violate the sampling condition with adjacent regular cells. Notice that the regular cell on the right side of the bridge cell of Figure 4.3 must be reconstructed with an intra-cell vertex, rendering configurations (c) and (f) invalid. In our example, both bridges having three signature chains are left with three valid configurations only, reducing the number of potential reconstructions to 288.

Lastly, we follow the procedure described in Section 2.2.4 and traverse all potential reconstructions to find those that are consistent with the input and comply with
the sampling condition. In our example, there are only four non-self-intersecting reconstructions. If one wishes to generate reconstructions with trimmed corners, there are eight such reconstructions. Figures 4.4 and 4.5 show the two respective sets of solutions. Notice that the first four reconstructions in Figure 4.5 are the same as the four reconstructions in Figures 4.4, having noncompulsory corners trimmed. The other four reconstructions are due to the relaxation of allowing polygon vertices at the endpoints of the intersection segments.
4.2 More Results

During the execution of the algorithm we gather several statistics, such as the complexity of the input, the number of cells in the data structure, and the number of potential reconstructions (before and after reduction). For each valid reconstruction, we output additional statistics, including the number of vertices, the number of polygons and holes comprising the reconstruction, the overall perimeter, and the roughness of the reconstructed polygons as the average deviation of dihedral angles from 180°. We also record the net run time of the algorithm (on a PC with Intel P4 2.4GHz processor with 0.5GB RAM), excluding rendering time.

Figure 4.6 presents four additional examples we experimented with (the input polygon(s) that generated the input appear dashed), and the following table shows complexity-related statistics for these four test cases, for non-self-intersecting reconstructions.
Figure 4.5: Eight reconstructions with trimmed corners
<table>
<thead>
<tr>
<th>Case</th>
<th>$\ell$</th>
<th>$s$</th>
<th>Bridges</th>
<th>Combinations</th>
<th>Reduced comb.</th>
<th>Recons.</th>
<th>Time(sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>17</td>
<td>50</td>
<td>15</td>
<td>16,384,000</td>
<td>512,000</td>
<td>88</td>
<td>22</td>
</tr>
<tr>
<td>(b)</td>
<td>2</td>
<td>8</td>
<td>3</td>
<td>280,280</td>
<td>280,280</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>(c)</td>
<td>9</td>
<td>20</td>
<td>9</td>
<td>250,000</td>
<td>75,000</td>
<td>22</td>
<td>1.5</td>
</tr>
<tr>
<td>(d)</td>
<td>50</td>
<td>51</td>
<td>10</td>
<td>5120</td>
<td>1536</td>
<td>6</td>
<td>—</td>
</tr>
</tbody>
</table>

Figure 4.6: Four test cases
We show additional statistics for case (a), comparing the output in three modes of operation:

1. Self intersecting reconstructions;

2. Non-self-intersecting reconstructions; and


Notice that in the third mode of operation, the algorithm does not attempt to create intra-cell vertices unless they were previously tagged as compulsory, hence, the run time is greatly reduced.

<table>
<thead>
<tr>
<th>Mode</th>
<th>nVertices</th>
<th>Perimeter (units)</th>
<th>Roughness (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>median</td>
<td>std</td>
<td>median</td>
</tr>
<tr>
<td>(1)</td>
<td>33</td>
<td>3.6</td>
<td>2009</td>
</tr>
<tr>
<td>(2)</td>
<td>32</td>
<td>3.5</td>
<td>1912</td>
</tr>
<tr>
<td>(3)</td>
<td>45</td>
<td>1.8</td>
<td>2588</td>
</tr>
</tbody>
</table>

4.3 Comparison with Previous Work

As described in the introduction, Coll and Sellarès [CS01] dealt with a problem that closely resembles ours. The input to their algorithm is a “dense” set $L$ of $\ell$ lines uniformly distributed over an unknown shape together with the ordered set of points in which each line intersects the boundary of the shape, namely, the endpoints of the intersection segments. The algorithm outputs, in $O(\ell \log \ell)$ expected run time, a triangulation from which a polygonal approximation, whose vertices are the endpoints of the intersection segments of the unknown shape, can be extracted in time proportional to the number of intersection segments.
The algorithm of Coll and Sellarès starts with a DCEL structure having an initial triangulation of the bounding box of the shape in a form of a frame, in which all triangles are classified as being outside of the shape. The main part of the algorithm processes the intersection segments that reside on the lines of $\mathcal{L}$ sequentially, and modifies the DCEL structure accordingly. When a new intersection segment is considered, the triangles that contain the endpoints of the segment are subdivided, and a region determined by a subset of the triangles crossed by the segment is re-triangulated and reclassified as outside or inside in order to connect two vertices by an edge. Notice that if an intersection segment falls within triangles that are classified as outside, only the new half edges are classified as inside, and not the triangles themselves. Eventually, the set of triangles that are classified as inside compose the approximation of the shape. Figure 4.7 illustrates the above.

Notice that the algorithm of [CS01] outputs a single reconstruction and that the vertices of the reconstructed polygon(s) are the endpoints of the intersection segments. Our algorithm, on the other hand, outputs all possible reconstructions that are consistent with the input and forces the vertices of the reconstructed polygon to be the endpoints of the intersection segments only if solutions with trimmed corners are sought.

We return to the walk-through example of Section 4.1 and compare in Figure 4.8 the reconstruction obtained by the algorithm of Coll and Sellarès with the reconstruction produced by our algorithm (with trimmed corners) that resembles theirs the most (rather subjectively). Notice that due to the nature of their algorithm, isolated intersection segments do not contribute to the reconstruction, and that their algorithm is incapable of bridging gaps between groups of connected intersection segments.
(a) Boundary edges crossed by an intersection segment

(b) Connecting the end vertices of the boundary edges

(c) Inside classification of the faces containing the segment

Figure 4.7: Illustration of the algorithm of Coll and Sellarès

(a) Coll’s reconstruction  (b) Reconstruction of Figure 4.5(h)

Figure 4.8: The reconstruction of Coll and Sellarès vs. ours
In order to compare the run time of both algorithms, we present a degenerate version of our algorithm that, although not incremental in nature, provides a single reconstruction that is identical to the one created by the algorithm of Coll and Sellarès and is comparable in its run time.

We modify our algorithm to ignore all signature chains that consist of a single signature segment, and restrict the bridge cells to a configuration in which each signature chain is connected to itself. Thus, we are forcing a single reconstruction that is composed only of vertices that are the endpoints of the intersection segments, which is identical to the reconstruction of Coll and Sellarès.

The run time of this degenerate version of our algorithm is dominated by the time needed to initialize the data structure \(O(\ell^2 + s)\) in the worst case, see Section 3), and the time for generating the reconstruction, which is \(O(s)\). The total run time is, therefore, \(O(\ell^2 + s)\) in the worst case, which is slightly worse, but still comparable to that of the algorithm of Coll and Sellarès, which is \(O(\ell \log \ell + s)\) expected time. To conclude the comparison to Coll’s algorithm, we have shown that even though our algorithm was not conceived with the intention of producing a single solution, it may be still used in its degenerate version to produce a solution identical to that of Coll, with a comparable run time complexity.
Chapter 5

Concluding Remarks

In this thesis we investigate the problem of two-dimensional polygon reconstruction from line cross-sections. We focus on, but do not limit ourselves to, cases where the number of cutting lines is small and seek all reconstructions that are consistent with the input and comply with a natural sampling condition. We describe an algorithm for providing these reconstructions in detail.

Our complexity analysis shows that the number of reconstructions, and hence, the run time of the algorithm, is exponential in nature, since it involves products of Catalan numbers. If we restrict the algorithm to provide a single reconstruction, identical to that of [BCL96], the run time is quadratic in the size of the input. The related question of conditions for uniqueness of the solution remains open.

Further study may address the three-dimensional counterpart of the problem treated here, namely, three dimensional reconstruction of a polyhedron from planar cross-sections. There is a clear analogy between the two problems, where the signature chains on the boundary of the two-dimensional cells generalize to signature patches on the facets of the three-dimensional cells. These patches are connected at their boundaries by triangles, to create the polyhedral mesh.
Bibliography


 naprawתי וICATIONS עם מילים

אלהרגיון מומש ב-C++, המשמש ב-C++

Visual-Studio .Net, C++ ממילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון وלחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ ומילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,

אלהרגיון מומש ב-C++ למילא, וJKLMEScher2D VTK

בנ שרי חיתוך פוליגון ולחיתוך,
şişinat הלאגורים

בעבדה זו הראו כי מספר התרצורות של ה-\( n \) נבדל בין显示器 ש.setInt n `Catalan` משולש הכוכב, ו-\( C_n \) משולש הזווית ל-\( \phi \) עם השולחן של ה-\( \phi \). [CS01]

בכל \( n \) נצמד显示器 הלאגורים ל-\( \phi \) בחודשduğu של显示器 ב-\( \phi \). [CS01]

כמו בתוספת משולש הכוכב, \( b+x \)显示器显示器 הכוכב ב-\( b \).
הטורת היא התאמת קוזותرشארות התוחנה עליידי מיתרים שיאנס
התושרחש

שוחור את: חיבור קוזותرشארות התוחנה עליידי קיוש או עלי
ידי של שירים בכדי קדקוד מ Fetish הנמקת בתוכו התא. שטס מספר
שוחרים להאודה אתונה של התא.

שוחר הפולטי: שוחר את התא הנבול בצירוף תנוע של התוחנה שלחה.

הטור את המרכז של התא间距ים או שוחר מבצעים התוחנה של פולטים
בכל התא נבל. العراעilmiş בחלק השערני של התואים על ממט לייצר שוחרים
שעות עקביים על כל התוכניים או התא התוכנית. שוחר פועשה עבד
כל יצורים אפשריו של התוחנה התאיא.

הalarerrios מ المركز משולב向き:

• אתרחל בטיס התוכנית;
• **********************************************
• עטוף מספי התוארות על התאнего. ר.
• **********************************************

 şiיר התוחנה התוכנית לכל יצירת מבנה של
המכיל את התאיאי,
וייס מחנות שלם, או תשרירת התוחנה על הקפס. הבאים סופיים
הטור מבצע התוחנה על התוכניים המ网站地图 שלם קדימה
ל[idx], התוכניים מസוגים על התוכניים התוכניים (שאינו לוגלטס לוגר
הalarerrios) ו Crate גבלי (חרים ריגליים חטיא גורא).
לפי되었다 (אווד ואוור) \( P \) יש לכל ואט כל הש런損害 \( R \)

\[ R \cap L = P \cap L = S \]

על כל תבלי מוסף שסי פתרונח ניסחי תאני \( W \) הפחותי לכל

על בשחוון פלונית \( R \) הפחות על-ידי שימי \( W \) הפחות במקומם

הנאי הזדמנות מאפסים את מוספר הפתרונח האפשריים בכל \( W \)

מכסה פתרון \( W \).

במסגרת עבורי דו ממסונף \( P \) הפוחר את בטיע השחרור שחרור

לעיל מובאים לכל מוספר מושג פתרונח \( L \) מוסק הזרוע השחרורי:

1. חא: חלק קומור של התנעור התנועה על \( P \) ממקסימ ושמיר \( W \).
2. אט מוסיספים מבלי חוסם כל של התראים היה סופיים.
3. תמישת תיגור: חלק של כסה ח.orange השוכן על \( W \) חות מוסכים.
4. שערית תיגור: שערור ממקסימ התנועה המגנינא בקצותיה. שערית
5. התנועה הרביעית ממקסימ אוחז פערות בחוך על מוספר קTelefono.
6. של ה\( W \).

חא גל: ה\( W \) המכל ה\( L \) פערות שערית התנועה \( W \)

אם התשרשות מחווה את \( W \) וחרם כל \( W \) מסווים מתא פנים.娅פ

קול בatorio פלונית. ה\( W \) שיאפ ממקל שערית התנועה \( W \) מסווים מתא

ה\( W \).

חא גל: ה\( W \) בורל בעלא שערית התנועה \( W \)

חא גל: ה\( W \) גלא בעלא \( W \) שערית התנועה \( W \)
 suscepts analyzes process exposure and exposure levels of inflammatory cytokines and inflammatory mediators.


Conclusions: The results of the present study suggest that the inflammatory response in the hypothalamic region following thermal injury may be mediated by the release of pro-inflammatory cytokines and mediators, which could contribute to the development of hypothalamic dysfunction and consequent complications.

The conclusions of this study are based on the following observations:

1. The inflammatory response in the hypothalamic region following thermal injury is mediated by the release of pro-inflammatory cytokines and mediators.

2. The inflammatory mediators released in the hypothalamic region following thermal injury may contribute to the development of hypothalamic dysfunction and consequent complications.

3. The inflammatory response in the hypothalamic region following thermal injury may be modulated by the release of anti-inflammatory cytokines and mediators.

In conclusion, the inflammatory response in the hypothalamic region following thermal injury is an important factor in the development of hypothalamic dysfunction and consequent complications.

References:

<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>4.4</td>
</tr>
<tr>
<td>31</td>
<td>4.5</td>
</tr>
<tr>
<td>32</td>
<td>4.6</td>
</tr>
<tr>
<td>35</td>
<td>4.7</td>
</tr>
<tr>
<td>35</td>
<td>4.8</td>
</tr>
</tbody>
</table>

Sellarès' Coll
רשימת אオリים

8 .......................... דוגמה ממיתת לעיניית Şיוואר הפלנּון
9 .............................. תכרים על פניה של פלנּון
2.1
10 ............................. את דוגמה
11 ............................ סוף את
2.2
12 ........................... כל כהונפומריאות של גנרי עב שלום שלשרואט וחתימה
2.3
13 ............................... שחרור את
2.4
14 ................................. ממיתת של כהונדוק חובה
2.5
15 ............................... ביצוי כהונפומריאות של גנרי
2.6
16 ............................... קומביינויה לא חועד של כהונפומריאות של גנרי סמטוכים
2.7
17 ................................. שחרור אפשי של ערי גנרי סמטוכים
2.8
18 ................................. חיתון פנו
2.9
19 ................................. גיאנשייין כהומית שאריה ומשכרים
2.10
20 ................................. דוגמה סמיאפיט (חסמ התחנה) לבליטי הטרקיה
2.11
25 ................................. קול
2.12
27 ................................. אטרק הגרים והתא אגר מודיסים
2.13
28 ................................. קופה רפורט.charCodeAt(0) נורה קודקורי חובה
2.14
29 ................................. קופה רפורטCharCode(0) נורה קודקורי חובה
3.1
21 ניוגו סידברות

27 תשלומים נסיוניות

27 דוגמא מיניונית
30 תשלומים נsteryファン
33 הشهادה על עבידה קדמית

37 סיכום

39 רשימות מקורות
תוטם עיניים

תчкиיר

1  נבואה 1

2  שעון פוליגון

7  תדרות הבעיה .............................................. 2.1
8  .................................................................................. 2.1.1
9  תדרות הזדמנות ................................................. 2.2
9  .................................................................................. 2.2.1
12  איזור מבנה התנועה ......................................... 2.2.2
14  כונסגריניט פארם ............................................... 2.2.3
15  שעון הפוליגון ...................................................... 2.2.4
17  הƣוראות אלגוריתמים ......................................... 2.3
17  .................................................................................. 2.3.1
18  .................................................................................. 2.3.2
19  הƣוראות ממית ...................................................... 2.4
19  .................................................................................. 2.4.1
20  דיק גיאומטרים ..................................................... 2.4.2
המחק מתלהב בהבחין פרי פיפסוא ריבertia וה למה הפיסות חיות חסיך

םקללת למין המחשה.

ברא哮יו להותיון לחים וללי עלי כח שיקודא לעצק בנתך מחקר כמפעני,
על התייתו המסטור על כל המ שידעתי מרחס במקל העבורה.
כמו כה, אווד לחים ללי על התלות האמירה לקוח ואחרי תחתיוスピות
ברוחנין סטודוט הילוינ.
חוף פלוגות מחטלים קיימים

היבר על מחקר

לש מיכלי חלקי של הדרישות לקבלי החאר

מיגש על המדען במודיע המוחשב

אבישי סנלרסקי

הובש לסנט הטכניון - מכון טכניולוג יישראלי

חיפה

apus 2006

נסל תשט”ו
שוזור פליגון מחטביים קיימים

איביש סידני

?