Tools, Generic Programming, CGAL, and 2D Arrangement of CGAL

Efi Fogel

efif@post.tau.ac.il

School of computer science, Tel Aviv University
Tools, Toolkits, and Libraries

Computational Geometry
- LEDA - combinatorial and generic data types and algorithms
- CGAL - computational geometry generic data types and algorithms
- GMP & CORE - Numerical types
- Other

Visualization
- OpenGL, Direct3d
- GLUT
- QT
- Geomview
STL - Standard Template Library

first example of generic programming, 1994

A simple example

```cpp
int main() {
    vector<string> V;
    string tmp;

    while (getline(cin, tmp))
        V.push_back(tmp);

    sort(V.begin(), V.end());
    copy(V.begin(), V.end(), ostream_iterator<string>(cout, "\n"));
}
```
CGAL - Computational Geometry Algorithm and Database

- A Computational Geometry Algorithms Library
- A collection of Data Structure and Operations
- Consists of 3 major parts:
  - **kernel** constant-size non-modifiable geometric primitive objects and operations on these objects
  - **basic library** - a collection of basic geometric data structures and algorithms
  - **other facilities** - non-geometric support facilities, such as circulators, random sources, I/O support for debugging and for interfacing CGAL to various visualization tools
Written in C++
Follows the Generic Programming paradigm
Developed by a consortium consisting of
- ETH Zürich (Switzerland)
- Freie Universität Berlin (Germany)
- Martin-Luther-Universität Halle-Wittenberg
- Max-Planck Institut für Informatik, Saarbrücken
- INRIA Sophia-Antipolis (France)
- Tel-Aviv University (Israel)
- Utrecht University (The Netherlands)

CGAL home page - http://www.cgal.org
CGAL at TAU - http://www.cs.tau.ac.il/CGAL
CGAL - Kernel

- Contains objects of constant size
  - point, vector, direction, line, ray, segment, triangle, iso-oriented rectangle and tetrahedron.
  - Each type comes with a set of functions that can be applied to an object of this type.

- Access functions, tests of the position of a point relative to the object, a function returning the bounding box, the length, or the area of an object, etc..
  
  - More basic operations

- Affine transformations, detection and computation of intersections, and distance computations.
The collection of basic geometric algorithms and data structures
- polygons, half-edge data structures, polyhedral surfaces, planar maps, triangulations, convex hulls, alpha shapes, optimisation algorithms, dynamic point sets for geometric queries, and multidimensional search trees.

Tel-Aviv Planar Map Packages:
- **Topological map** - combinatorial information and no geometric information
- **Planar Map** - embedding of a topological map into the plane
- **Arrangement** - A structure that supports 2D arrangements. Maintains a planar map and curve hierarchy trees.
Robustness

Most geometric algorithms assume exact computation with real numbers.

Implementation of geometric algorithms becomes problematic.

Often the exact real arithmetic is replaced by inexact floating-point arithmetic in the implementation.

Rounding errors introduced by an inaccurate arithmetic may lead to inconsistent decisions, causing unexpected failures.
In CGAL you can choose the underlying number types and arithmetic.

You can use different types of arithmetic simultaneously.

The choice can be easily changed.

You can choose between implementations with fast but occasionally inexact arithmetic and exact computation.
LEDA - Efficient Data types and Algorithms

- A C++ class library
- Provides algorithmic in-depth knowledge in the field of graph and network problems, geometric computations, combinatorial optimization and other.
- Used in application areas such as GIS, VLSI design, telecommunication, scheduling, traffic planning, computational biology, and computer-aided design.
- Offers algorithm building blocks such as graphs, sequences, dictionaries, trees, points, flows, matchings, segments, shortest paths, and more.
- Follows object-oriented generic prog. paradigm
- CGAL & LEDA are independent, but work well together
- LEDA - http://www.algorithmic-solutions.de/redirected.htm
GNU MP is a library for arbitrary precision arithmetic operating on signed integers, rational numbers, and floating point numbers.

It has a rich set of functions, and the functions have a regular interface.

GNU MP is designed to be as fast as possible for small and huge operands.


CORE - Robust Numerical and Geometric computation

based on a novel number core, an API that defines four levels of numerical accuracies:

- Machine Accuracy (IEEE standard)
- Arbitrary Accuracy
- Guaranteed Accuracy
- Mixed Accuracy

Core home page - http://www.cs.nyu.edu/exact/core
GLUT & GLUI - Graphics Library Utility Toolkits

- **GLUT**
  - The OpenGL Utility Toolkit
  - An ANSI C implementation of GLUT for the X Window System -
    http://www.xmission.com/nate/glut.html

- **GLUI**
  - GLUT-based C++ user interface library
  - Window-system independent
  - provides controls (e.g., buttons, checkboxes, radio buttons, and spinners) to OpenGL applications.
Qt - GUI software toolkit/widget-kit

- Multiplatform - Windows, Linux/Unix, Mac OS X, and embedded Linux
- Object-oriented C++ application framework
- KDE is based on Qt
Planar graphs that are embedded in the plane
Given a collection $\Gamma$ of planar curves, the arrangement $\mathcal{A}(\Gamma)$ is the partition of the plane into vertices, edges and faces induced by the curves of $\Gamma$. 
The Package in Brief

Goal: construct, maintain, modify, traverse, query and present subdivisions of the plane

- Robust
- Generic
- Handles all degeneracies
- Efficient
- Easy to interface
Hierarchy

- **Topological_map**
  - Maintains topological maps of finite edges

- **Planar_map_2**
  - Maintains planar maps of interior-disjoint x-monotone curves

- **Planar_map_with_intersections_2**
  - Maintains planar maps of general curves (may intersect, may be non-x-monotone)

- **Arrangement_2**
  - Maintains planar maps of intersecting curves along with curve history
Planar Map Functionality

- Creation and Destruction
- I/O
  - Save, load, print (ASCII streams)
  - Draw (graphic streams)
- Modification
  - Insertion, removal, split, merge
- Traversal
- Queries
  - Number of vertices, halfedges, and faces
  - Is Point in Face
  - Point location - various strategies
  - Vertical ray shoot
Planar Map Traits

- Geometric interface
- Parameter of package
  - Defines the family of curves in interest
  - Package can be used with any family of curves for which a traits class is supplied
- Aggregate
  - geometric types (points, curves)
  - Operations over types (accessors, predicates, constructors)
Planar Map Supplied Traits

- **Segments**
  - Standard
  - Cached contains the underline line and flags

- **Polylines**

- **Conic arcs** (segments of circles, ellipses, parabolas, hyperbolas, and lines)
Planar Map Insertion Operations

- Incremental insert
- Aggregate insert
- Often information is known in advance
  - Containing face
    - Insert in face interior
  - Incident vertices
    - Insert from vertex, between vertices
  - Order around vertex
    - Insert from halfedge target, between halfedge targets
Planar Map Aggregate Insertion Operations

- Inserts a container into the map

```c++
template <class curve_iterator>
Halfedge_iterator
insert(const curve_iterator & begin,
       const curve_iterator & end,
       Change_notification * en = NULL);
```

- Two versions
  - Simplified - planar map no intersections
  - General - planar map with intersections

- Sweep based