Code Flexibility and Program Efficiency by Genericity: Improving CGAL's Arrangements

Efi Fogel, Ron Wein, and Dan Halperin

{efif,wein,danha}@post.tau.ac.il.

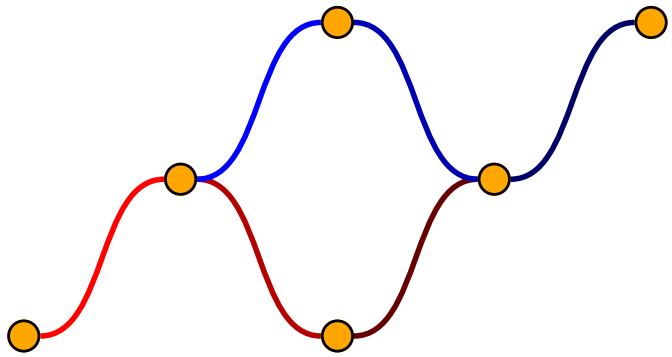
School of Computer Science, Tel Aviv University

Outline

- Introduction
- Innovations
  - Traits requirements and tags
  - Segment and Polyline traits
  - Data traits
- Demo

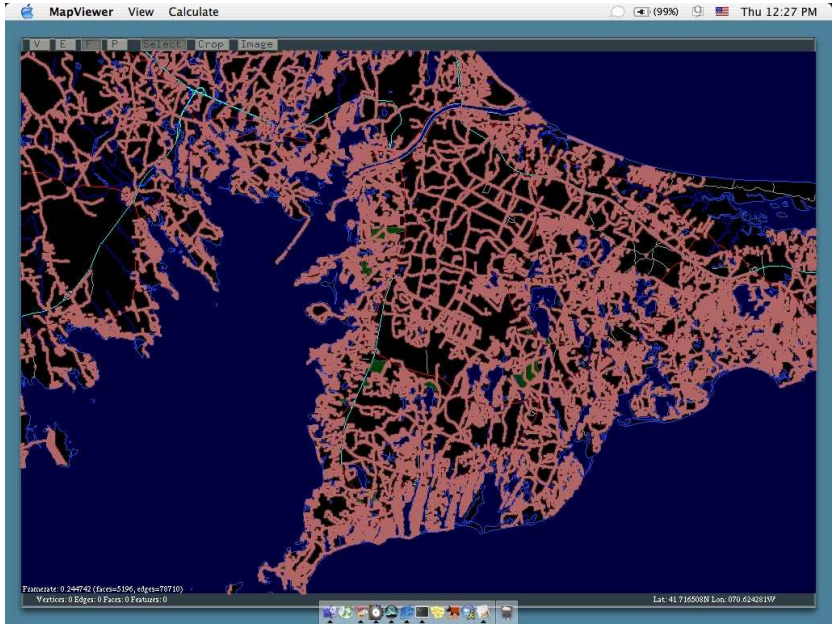
Planar Arrangements

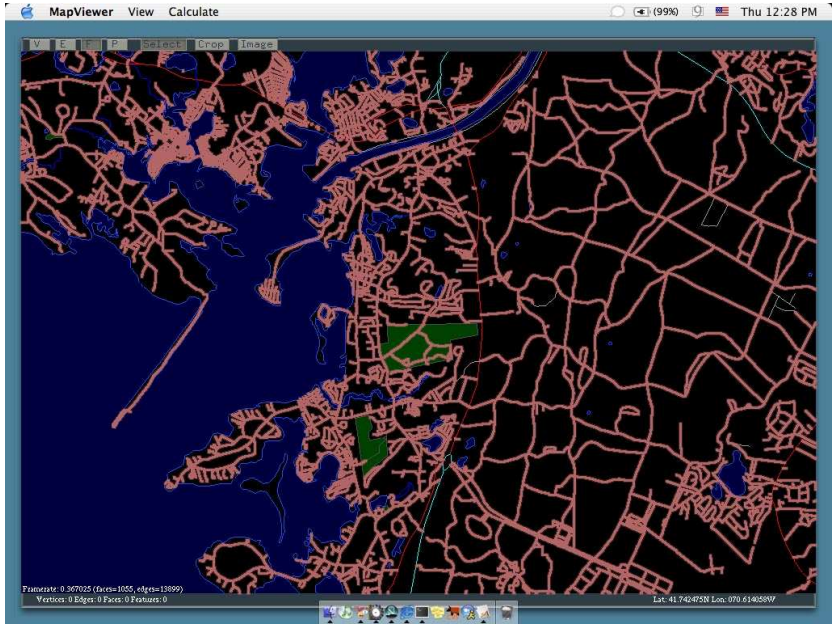
Given a collection Γ of planar curves, the arrangement $\mathcal{A}(\Gamma)$ is the partition of the plane into vertices, edges and faces induced by the curves of Γ



Background

- Arrangements have numerous applications
  - robot motion planning, computer vision, GIS, optimization, computational molecular biology

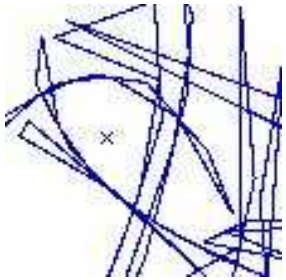
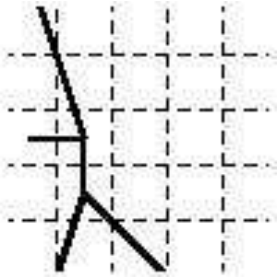
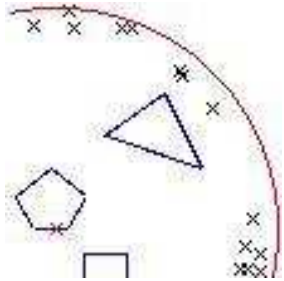
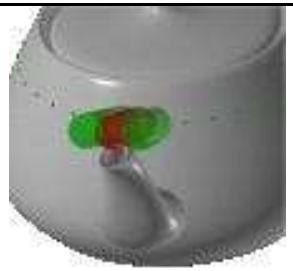
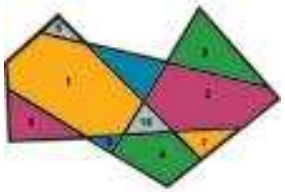
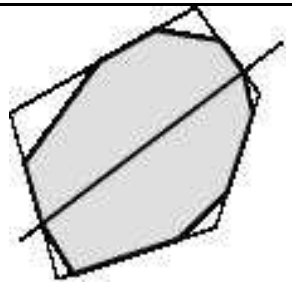
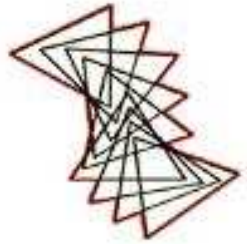




A CGAL planar map of the Boston area showing the top of the arm of cape cod.

Raw data comes from the US Census 2000 TIGER/line data files.

Applications of CGAL Arrangements

				
Adaptive point location	Iterated Snap Rounding	Minkowski Sums	Minimum Enclosing Disc	Hybrid Motion Planning
				
Path Verification	Inner-cover	A Map Edit Tool	Maximizing the Area	Union of Objects

http://www.cs.tau.ac.il/CGAL/Projects/

Goals

- Construct, maintain, modify, traverse, query and present subdivisions of the plane
- Robust, handles all degeneracies, exact
- Efficient
- Generic, easy to interface, extend, and adapt
- Conforms to CGAL (Computational Geometry Algorithms Library)

The CGAL kernel

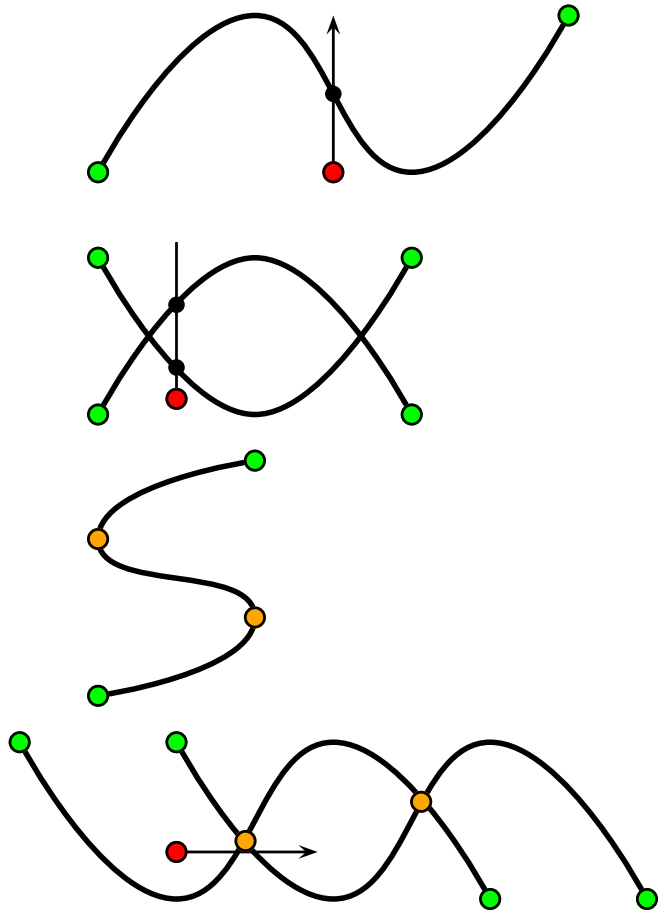
- CGAL provides a geometric *kernel* that consists of:
  - Geometric primitive objects (e.g., points, segments)
  - Predicates and operations on these objects

Planar Map Traits

- Geometric interface
  - Topology and geometry are separated
- 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 (point, curve)
  - Operations over types (accessors, predicates, constructors)

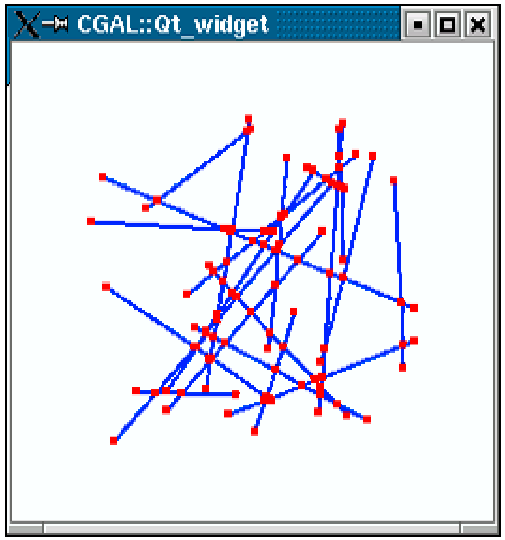
Traits Concept-requirements

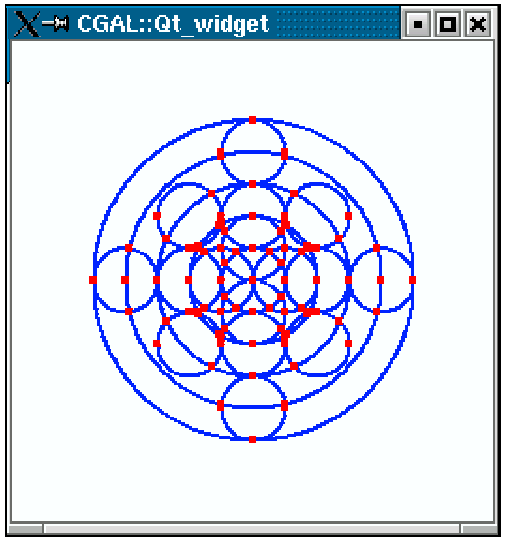
- Types: (**`Curve_2`**, **`X_monotone_curve_2`**, **`Point_2`**)
- Methods:
  1. Compare two points
  2. Determine the relative position of a point and a curve
  3. Determine the relative position of two curves at x
  4. Subdivide a curve into x-monotone curves
  5. Determine the relative position of two curves to the right (left) of a point
  6. Find the intersection of two curves to the right (left) of a point

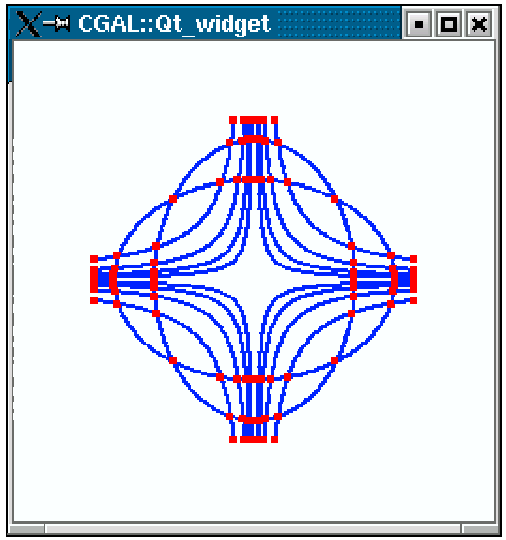


Planar Map Supplied Traits

- Segments
  - Kernel — represented by two end points only
  - Cached — contains the underlying line and flags
- Polylines
- Conic arcs (segments of circles, ellipses, parabolas, hyperbolas, and lines)





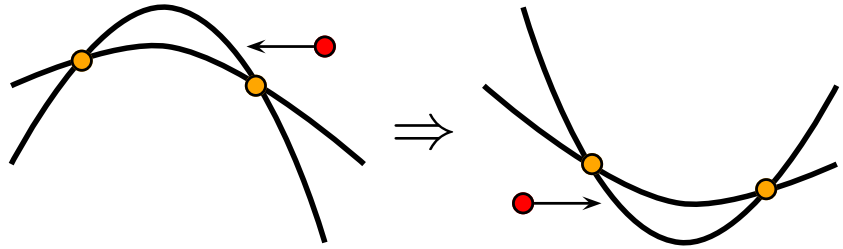


Planar Map Other Traits

- Bezier curves - Addaptive Approximation by Polygons
- Conics (Berberich et al., ESA 2002)
- Conics ("Curved Kernel", Emiris et al., SoCG 2004)
  - differentiated by algebraic computation-methods
- Cubics (Eigenwillig et al., SoCG 2004)

Tight and Flexible Traits

- Requirements are designed to be as tight as possible
- Comparing and finding the next intersection to the left of a point
  - are not required when constructing a planar map using the aggregate insert
    - The plane is swept from left to right
  - can be replaced by a pair of functions
    - that reflect a point or a curve about the axes

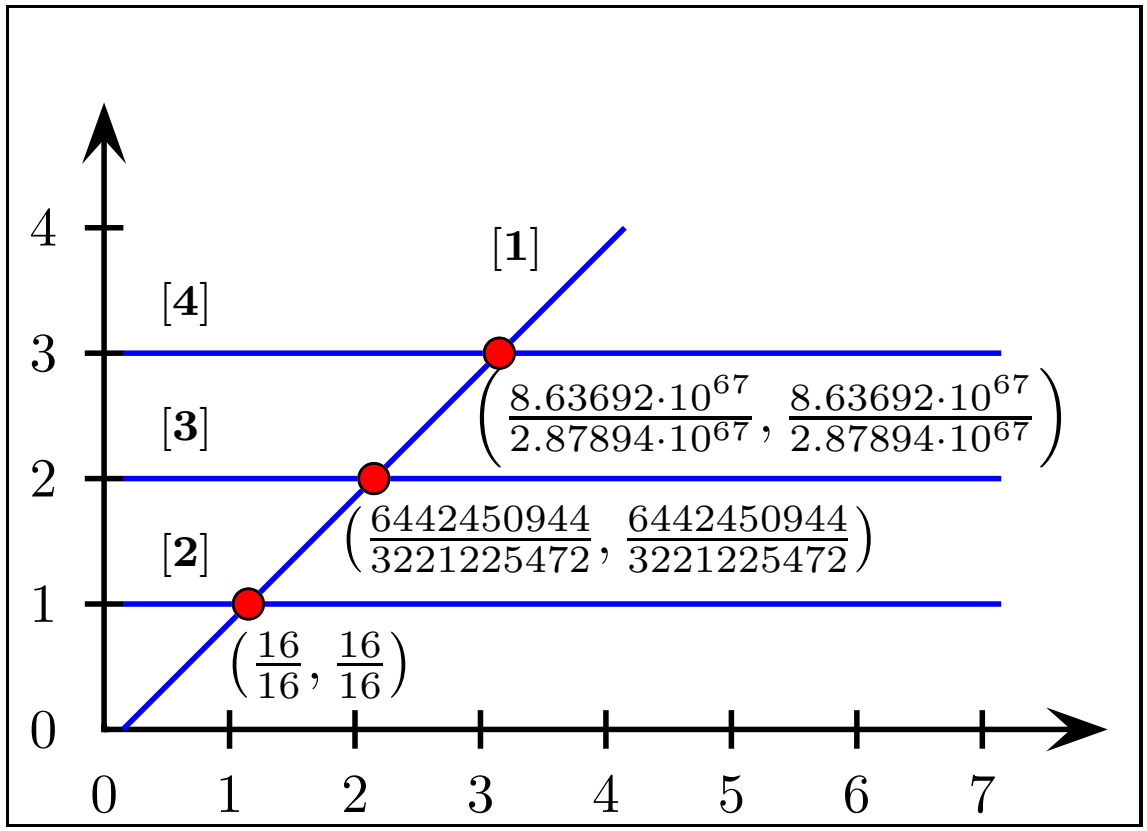


  - can be dropped all together, and implemented by the other methods
    - perhaps less efficiently

Kernel Segment Traits

`Arr_segment_traits_2`

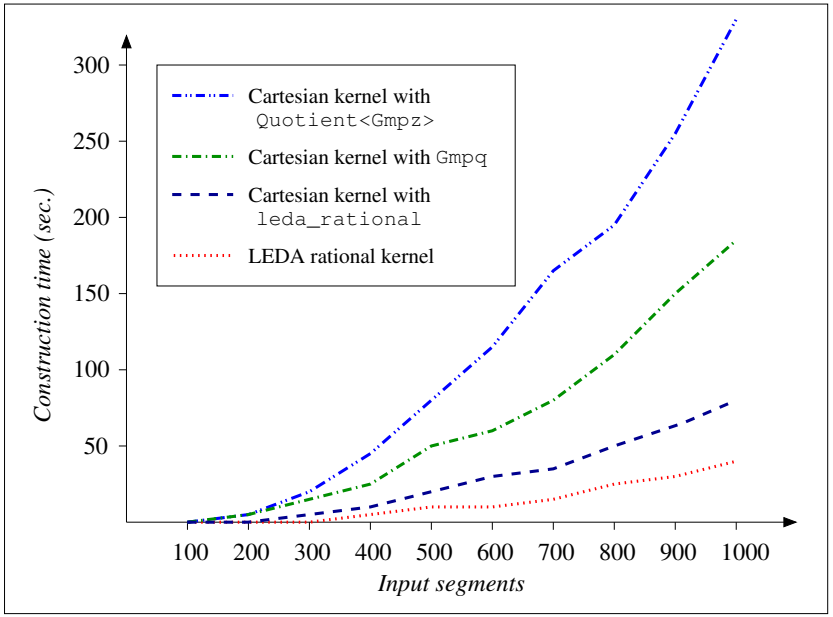
- Parameterized with geometric kernel
- A segment is represented by its 2 endpoints
- Leads to cascaded rep. of intersection points



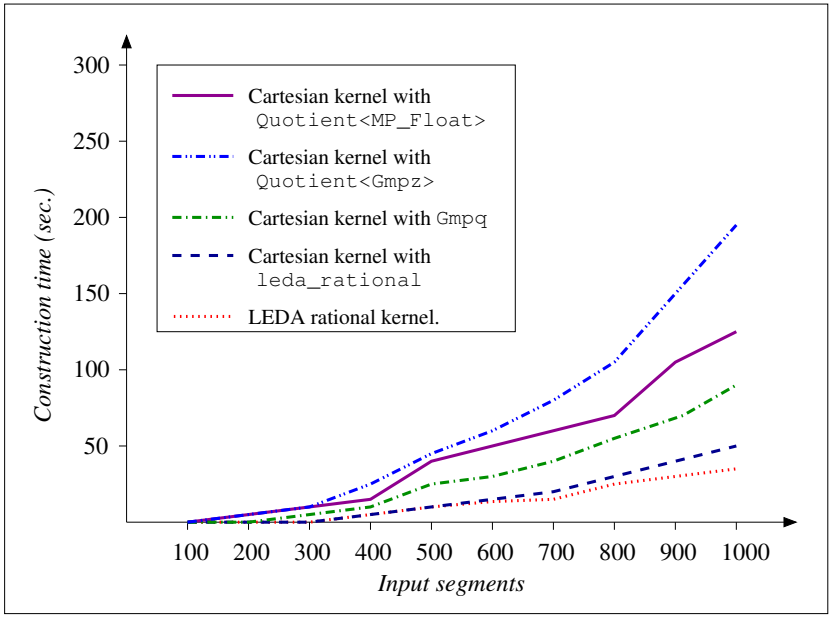
Cached Segment Traits
`Arr_segment_cached_traits_2`

- Parameterized with geometric kernel
- A segment is represented by its 2 endpoints and its underlying line
  - When a segment is split, the underlying line remains the same
  - The underlying line is used to compute intersections
- Faster than the kernel traits
  - Especially when constructing an arrangment with many intersections
- Indiscriminate normalization can be harmful

Experimental Results



Using kernel traits



Using cached traits

- Inserting 1000 random curves with **`Quotient<Qmpz>`** is $\sim 50\%$ faster with the cached traits vs. the kernel triats
- The LEDA rational kernel uses heuristics to normalize the numbers. Therefore, the speedup is minor
- The **`Quotient<MP_Float>`** number type cannot be used at all with the kernel traits, as its precision is limited

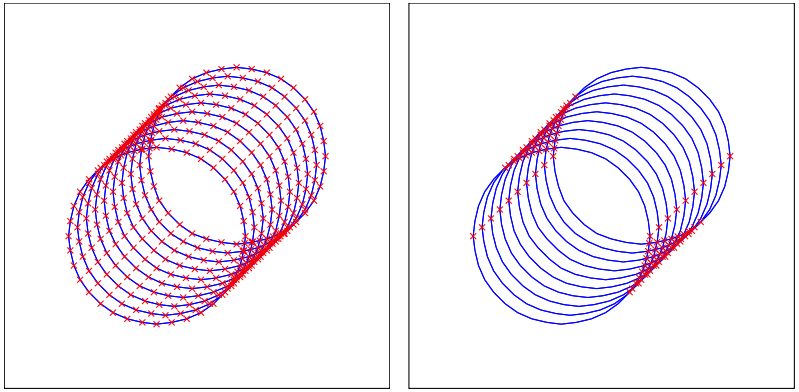
Planar map of Segments Type-Definition

```
typedef CGAL::Quotient<CGAL::MP_Float>                NT;
typedef CGAL::Cartesian<NT>                           Kernel;
typedef CGAL::Arr_segment_cached_traits_2<Kernel>     Traits;


typedef CGAL::Pm_default_dcel<Traits>                 Dcel;
typedef CGAL::Planar_map_2<Dcel,Traits>               Pm;
typedef CGAL::Planar_map_with_intersections_2<Pm>     Pmwx;
```

Polyline Traits

- Planar map of piecewise linear segments (polylines)
  - Approximate curves of high degree
  - Useful for testing
    - not necessarily x monotone
    - (Exact) rational number type is sufficient
- Parameterized with a segment traits
- Maintains a `Seg_traits::Curve_2` segment vector
- Underlying segment traits performs geometric ops.



Planar map of Polylines Type-Definition

```
typedef CGAL::Quotient<CGAL::MP_Float>                NT;
typedef CGAL::Cartesian<NT>                           Kernel;
typedef CGAL::Arr_segment_cached_traits_2<Kernel>     Seg_traits;
typedef CGAL::Arr_polyline_traits_2<Seg_traits>       Traits;

typedef CGAL::Pm_default_dcel<Traits>                 Dcel;
typedef CGAL::Planar_map_2<Dcel,Traits>               Pm;
typedef CGAL::Planar_map_with_intersections_2<Pm>     Pmwx;
```

Data Traits

- A meta traits
- Parameterized with:
  - A traits class — a model of the traits concept
  - A data class — contains extraneous data associated with a curve
- The Data is passed from curves to subcurves when
  - A curve is divided into x-monotone curves
  - A curve is split at an intersection point

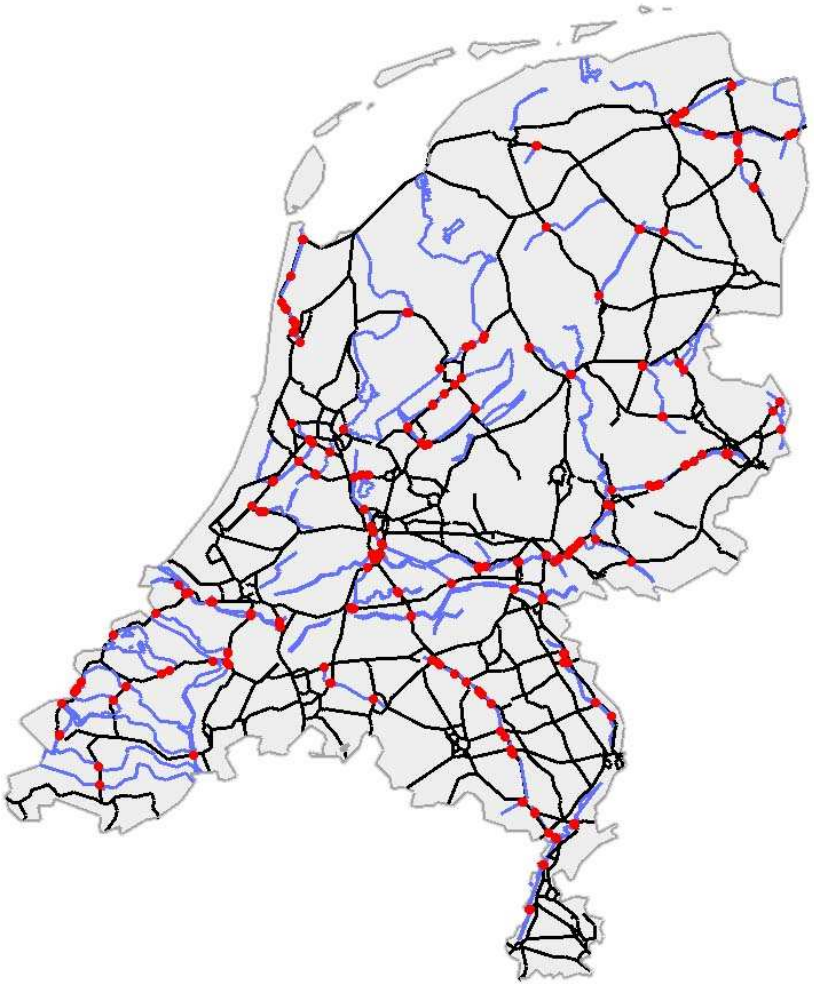
Applications of Data Traits

- A hierarchy of curves
  - Each curve contains a pointer to the curve it originated from
- An entire hierarchy of planar maps
  - Each x-monotone curve contains a pointer to an x-monotone curve in the planar map above in the hierarchy
- An overlay of rivers and roads

Planar map of Polylines with Data Type-Definition

```
struct Data {
  enum {ROAD, RIVER} type;
  std::string name;
};
typedef CGAL::Quotient<CGAL::MP_Float>                    NT;
typedef CGAL::Cartesian<NT>                               Kernel;
typedef CGAL::Arr_segment_cached_traits_2<Kernel>         Seg_traits;
typedef CGAL::Arr_polyline_traits_2<Seg_traits>           Bas_traits;
typedef CGAL::Arr_curve_data_traits_2<Bas_traits,Data>    Traits;
typedef CGAL::Pm_default_dcel<Traits>                     Dcel;
typedef CGAL::Planar_map_2<Dcel,Traits>                   Pm;
typedef CGAL::Planar_map_with_intersections_2<Pm>         Pmwx;
```

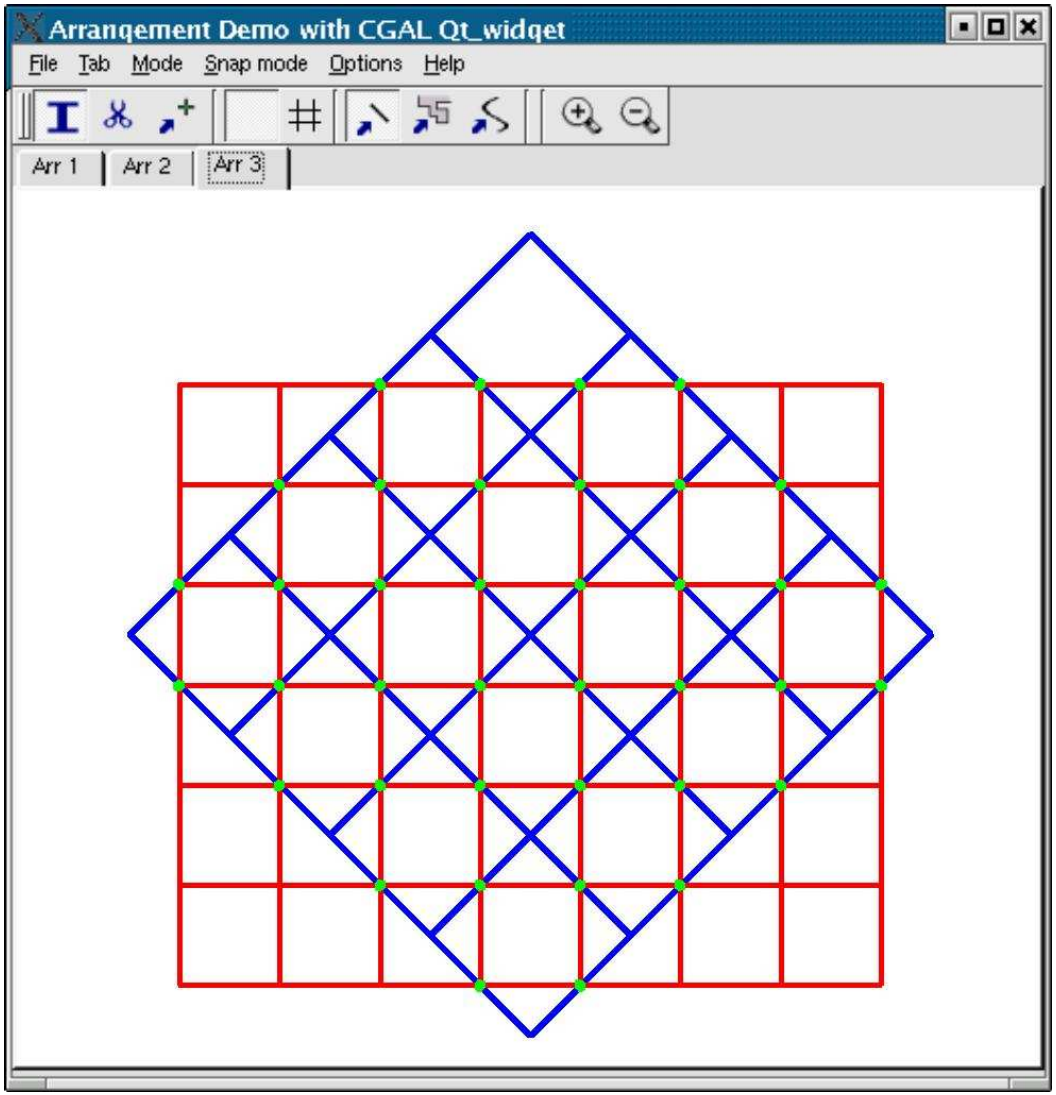
Example: the Netherlands



Roads, railroads, rivers and water canals on the map of the Netherlands. Bridges are marked by circles.

- border - 13 polylines, 373 segments
- streets - 877 polylines, 2360 segments
- water routes - 88 polylines, 1397 segments
- Takes less than 1 second to construct on Pentium 4 1.8GHz using `Gmpq` number type

Demo



by Niv Sabath