Problèmes de robustesse en géométrie algorithmique

Monique Teillaud

ENS - 12 mars 2008

Computational geometry

Solving geometric problems

- algorithms
- complexity analysis
  worst case, average, randomized
- implementation
- applications
  shape reconstruction, meshing, computer graphics, geographic information system, CAD, VLSI, structural biology...
  `http://www-sop.inria.fr/geometrica/`

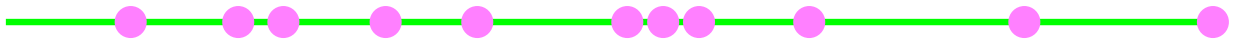
Implementing geometric algorithms

Ingredients for good software

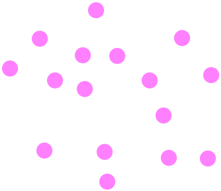
- clean mathematical formalism
- algorithmic study, data structures, complexity
- solving robustness issues
- good design and programming

First problem: convex hull
Definition

Dimension 1: sorting

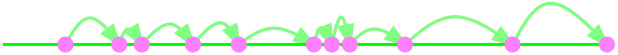


Dimension ≥ 2: convex hull

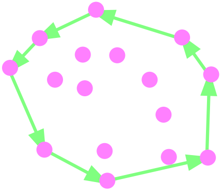


First problem: convex hull
Definition

Dimension 1: sorting



Dimension ≥ 2: convex hull



First problem: convex hull

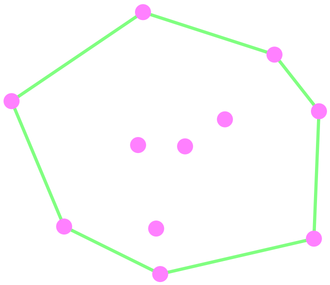
First problem: convex hull

predicate:

$$\text{compare}\,(x, y)$$

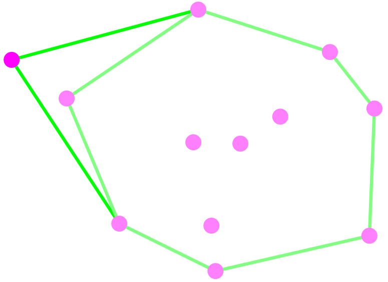
First problem: convex hull

Incremental construction

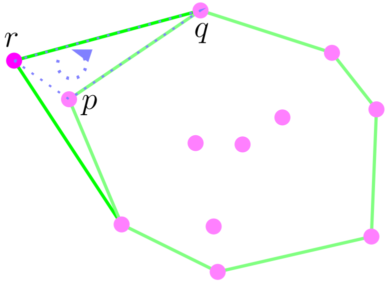


First problem: convex hull

Incremental construction



First problem: convex hull
Incremental construction



predicate:

orientation(p, q, r) =

$$\textit{sign}\left(\begin{vmatrix} 1 & 1 & 1 \\ p_x & q_x & r_x \\ p_y & q_y & r_y \end{vmatrix}\right)$$

(it is a resultant)

degree 2 polynomial

$$p = (0.5 + x.u,\ 0.5 + y.u)$$
$$0 \leq\ x, y\ < 256,\ \ u = 2^{-53}$$
$$q = (12, 12)$$
$$r = (24, 24)$$

$p = (0.5 + x.u,\ 0.5 + y.u)$
$0 \leq\ x, y\ < 256,\ \ u = 2^{-53}$
$q = (12, 12)$
$r = (24, 24)$

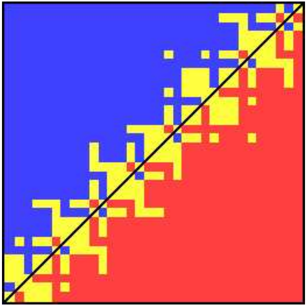
$orientation(p, q, r)$
evaluated with `double`

Orientation predicate
Arithmetic issues

$p = (0.5 + x.u,\ 0.5 + y.u)$
$0 \leq\ x, y\ < 256,\ \ u = 2^{-53}$
$q = (12, 12)$
$r = (24, 24)$

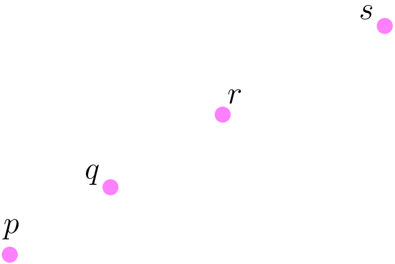
$orientation(p, q, r)$
evaluated with `double`

256 x 256 pixel image
> 0, $= 0$, < 0



Orientation predicate

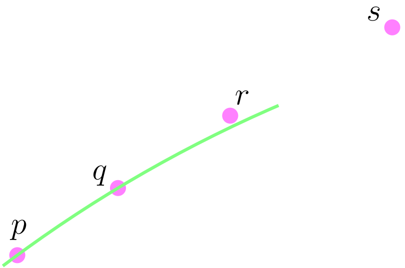
Arithmetic issues



$p <_x q <_x r <_x s$

Orientation predicate

Arithmetic issues

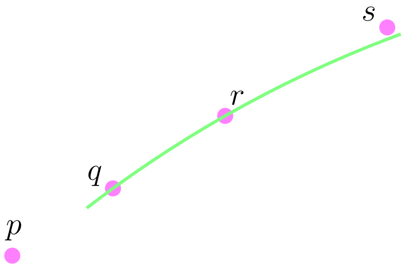


$p <_x q <_x r <_x s$

r above (pq)

Orientation predicate

Arithmetic issues

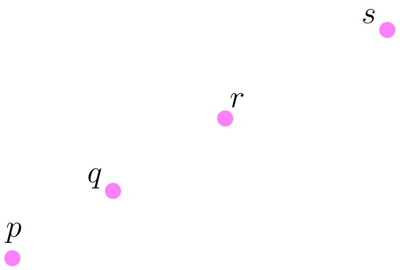


$p <_x q <_x r <_x s$

r above (pq)

s above (qr)

Orientation predicate
Arithmetic issues



$p <_x q <_x r <_x s$

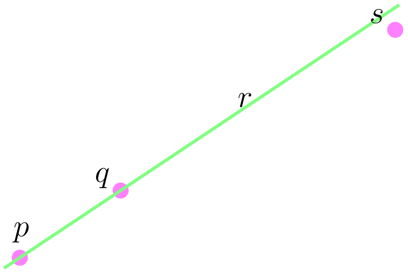
r above (pq)

s above (qr)

$\Longrightarrow s$ above (pq)

Orientation predicate

Arithmetic issues



$p <_x q <_x r <_x s$

r above (pq)

s above (qr)

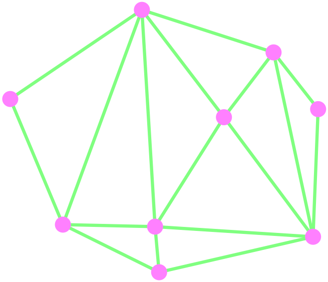
$\Longrightarrow s$ above (pq)

$\longrightarrow$ inconsistency in predicate evaluations

Two major data structures

- [Delaunay] triangulation
- Arrangement

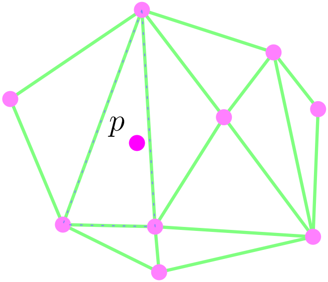
Triangulation
Incremental construction



For each new point p

Triangulation

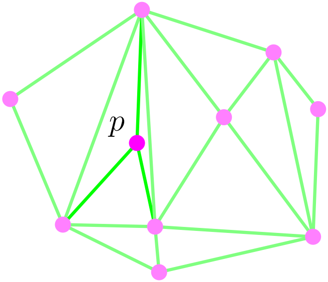
Incremental construction



For each new point p

- locate $p \longrightarrow$ triangle t

orientation

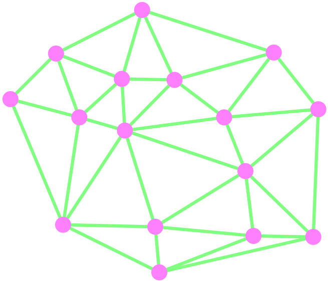


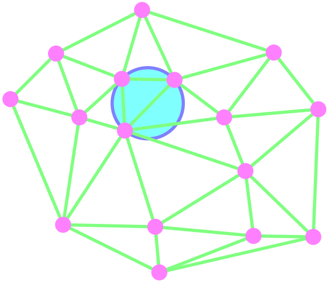
For each new point p

- locate $p \longrightarrow$ triangle t
- split t into 3 triangles

orientation

Delaunay triangulation
Definition

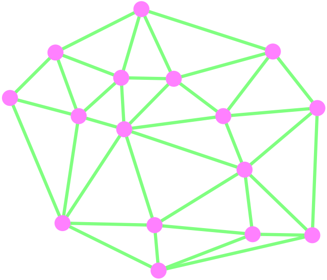




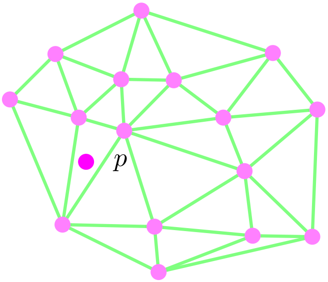
All circumscribing disks are empty

Dimension 2: Euler relation $n - e + f = 2 \rightarrow$ linear size
Dimension $d > 2$: size $\Theta\left(n^{\lceil \frac{d}{2} \rceil}\right)$

Delaunay triangulation
Incremental construction



For each new point p

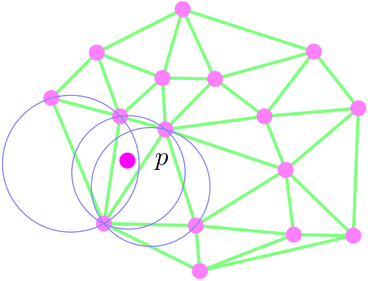


For each new point *p*

- locate *p* = find triangles in conflict

in_sphere

Delaunay triangulation
Incremental construction

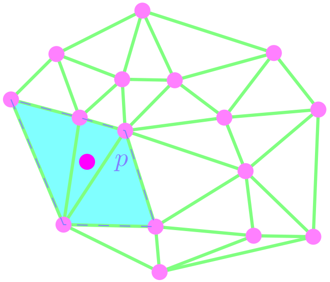


For each new point *p*

- locate *p* = find triangles in conflict   in_sphere

Delaunay triangulation

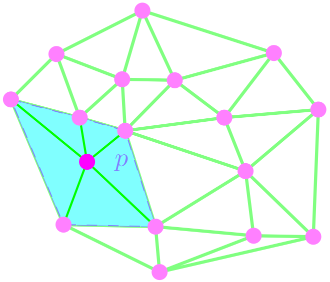
Incremental construction



For each new point p

- locate p = find triangles in conflict in_sphere

Delaunay triangulation
Incremental construction



For each new point p

- locate p = find triangles in conflict — in_sphere
- star the region around p

In_sphere predicate

$\text{in\_sphere}(p, q, r, s) =$

$$\mathit{sign}\left(\frac{\begin{vmatrix} 1 & 1 & 1 & 1 \\ p_x & q_x & r_x & s_x \\ p_y & q_y & r_y & s_y \\ 1+p_x^2+p_y^2 & 1+q_x^2+q_y^2 & 1+r_x^2+r_y^2 & 1+s_x^2+s_y^2 \end{vmatrix}}{\text{orientation}(p, q, r)} \right)$$

sign of degree 4 polynomial

In_sphere predicate

in_sphere(p, q, r, s) =

$$\textit{sign}\left(\frac{\begin{vmatrix} 1 & 1 & 1 & 1 \\ p_x & q_x & r_x & s_x \\ p_y & q_y & r_y & s_y \\ 1 + p_x^2 + p_y^2 & 1 + q_x^2 + q_y^2 & 1 + r_x^2 + r_y^2 & 1 + s_x^2 + s_y^2 \end{vmatrix}}{\text{orientation}(p, q, r)}\right)$$

sign of degree 4 polynomial

circumcenter/radius never computed

Exact Geometric Computation

$$\left.\begin{array}{l}\text{imprecise numerical evaluations}\\[1em] \text{combinatorial result}\end{array}\right. \quad \longrightarrow \quad \text{non-robustness}$$

Exact Geometric Computation

$$\left.\begin{array}{l}\text{imprecise numerical evaluations}\\[2em]\text{combinatorial result}\end{array}\right. \longrightarrow \quad \text{non-robustness}$$

Use of exact arithmetics

Evaluation of signs of polynomial expressions:
multiprecision rationals or floats

Exact Geometric Computation

imprecise numerical evaluations		
	$\longrightarrow$	non-robustness
combinatorial result		

Exact Geometric Computation

$\neq$

exact arithmetics

Optimize easy (frequent) cases

approximate computation
+
rounding errors controlled

Use exact arithmetics only on difficult cases

Cost $\simeq$ cost of floating point/double evaluation

Exact Geometric Computation

Filtering

Approximate evaluation $P^a(x)$
+ Error ε

$|P^a(x)| > \varepsilon$
?

y: $\operatorname{sign}(P(x)) = \operatorname{sign}(P^a(x))$

n: Exact computation

The Computational Geometry Algorithms Library

Open Source project

`www.cgal.org`

$>$ 400.000 lines of C++ code
$>$ 3.000 pages manual
$\sim$ 10.000 downloads per year
$\sim$ 850 users on public mailing list, $\sim$ 50 developers
LGPL, QPL
start-up GeometryFactory
interfaces: Python, Scilab

Robustness and efficiency

- Editorial board (3 members in Geometrica $\subset$ 11 members)
- Test-suites each night

...

Delaunay triangulations

CGAL-3.1-I-124

Pentium-M 1.7 GHz, 1GB
g++ 3.3.2, -O2 -DNDEBUG

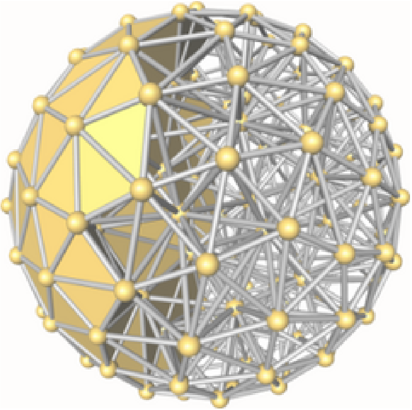
1.000.000 random points

double	48.1 sec
MP_Float	2980.2 sec
Filtered exact	58.4 sec

25 sec in release CGAL 3.3
(space filling curve)

CGAL-3.1-I-124

Pentium-M 1.7 GHz, 1GB
g++ 3.3.2, -O2 -DNDEBUG

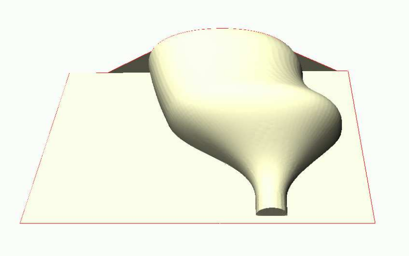


degeneracies explicitely handled
symbolic perturbations...

Delaunay triangulations

CGAL-3.1-I-124

Pentium-M 1.7 GHz, 1GB
g++ 3.3.2, -O2 -DNDEBUG

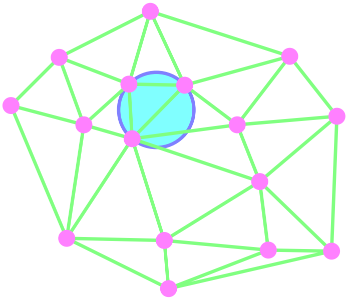


49.787 points
(Dassault Systèmes)

double	loop !
exact and filtered	$<$ 8 sec

Predicates and constructions

Delaunay triangulation

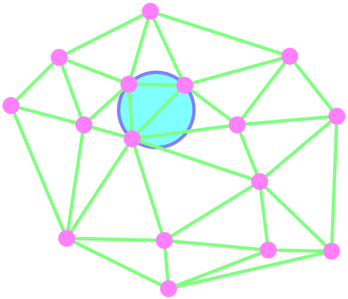


Only predicates:

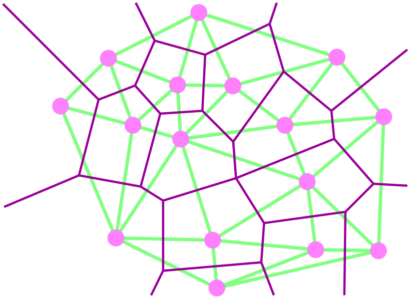
orientation, in_sphere

Predicates and constructions

Delaunay triangulation



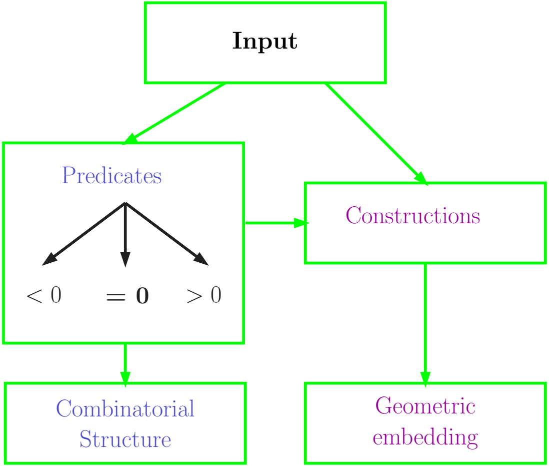
Voronoi diagram
geometric dual



Only predicates:
orientation, in_sphere

also constructions:
circumcenter

Predicates and constructions



Arrangements
Definition

Partition of the plane into

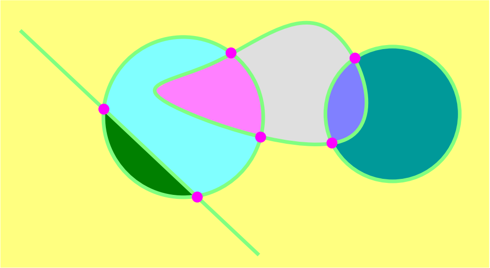
- faces
- edges
- vertices

induced by a collection of curves

Partition of the plane into

- faces
- edges
- vertices

induced by a collection of curves

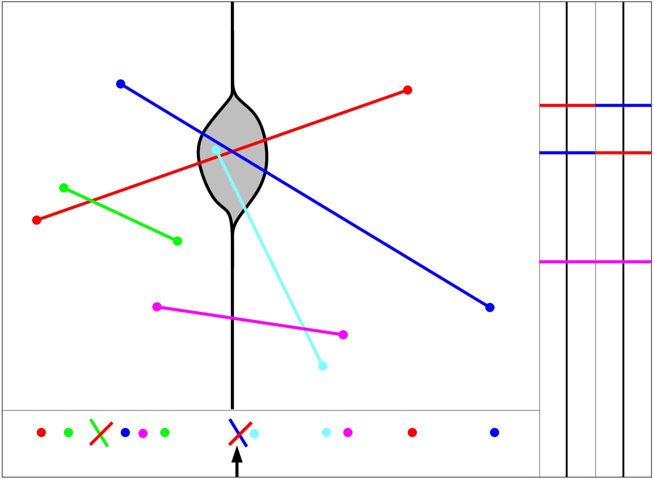


Bentley-Ottmann sweep

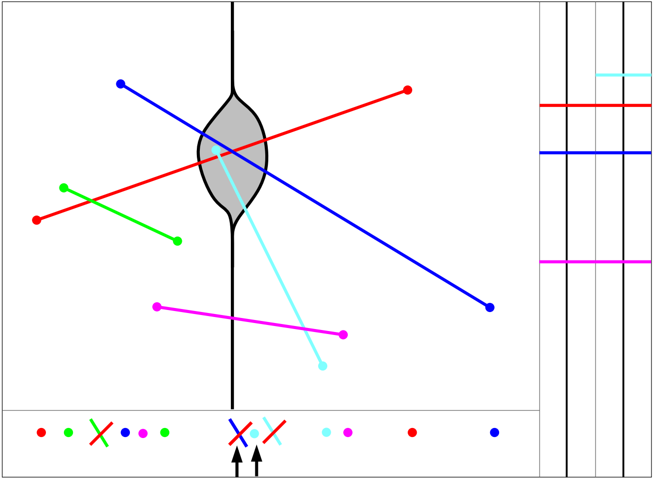
SLIDES

highly sensitive to arithmetic rounding

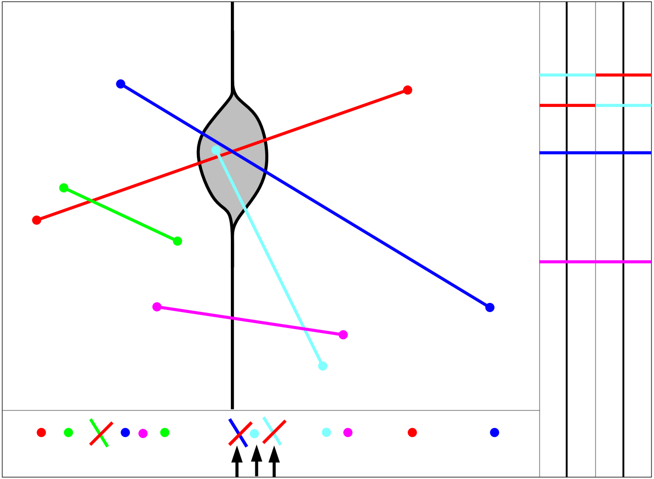
Wrong comparison $\longrightarrow$



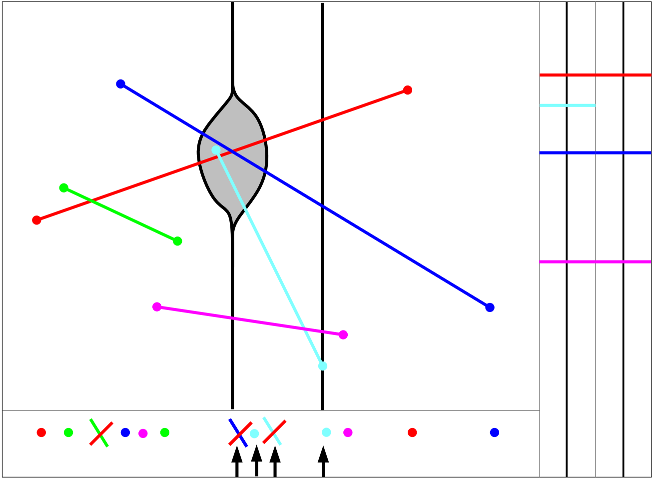
Wrong comparison $\longrightarrow$



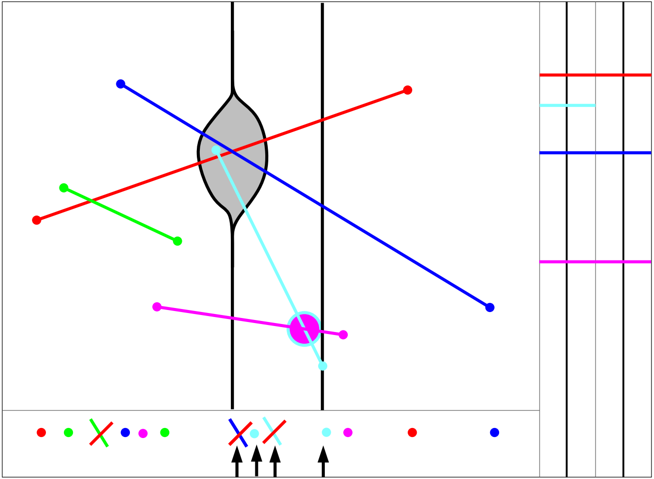
Wrong comparison $\longrightarrow$



Wrong comparison $\longrightarrow$

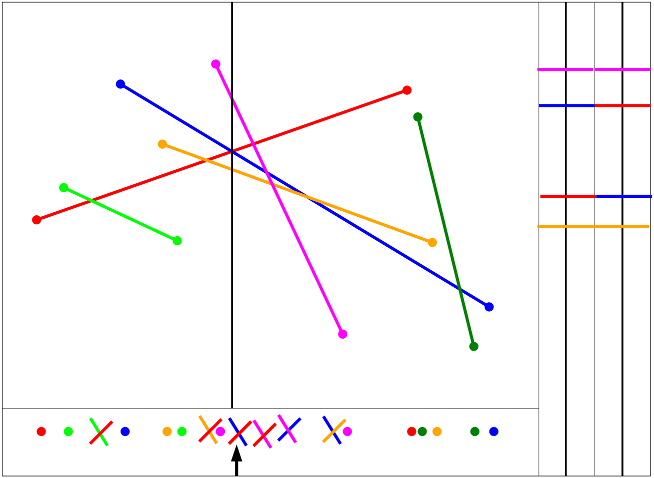


Wrong comparison $\longrightarrow$



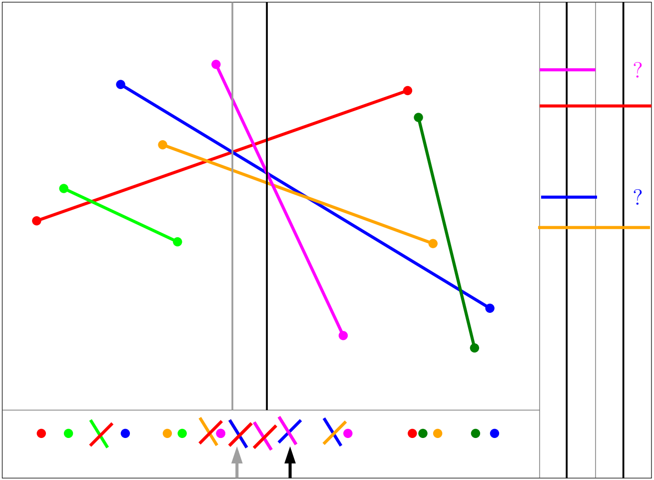
$\Longrightarrow$ intersection missed

Wrong comparison $\longrightarrow$



Arrangements
Arithmetic issues

Wrong comparison $\longrightarrow$



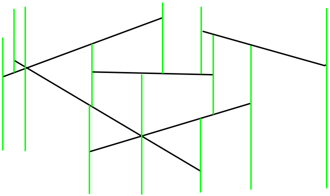
pink and blue are not consecutive $\Longrightarrow$ failure

S set of n segments in the plane

- 1st pb. Compute the pairs of segment that intersect
- 2nd pb. Compute the arrangement A

S set of n segments in the plane

- 1st pb. Compute the pairs of segment that intersect
- 2nd pb. Compute the arrangement A
- 3rd pb. Compute the trapezoidal map T



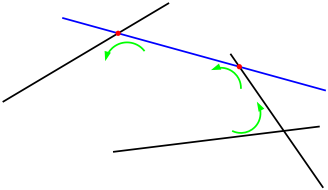
k= number of intersections

number of edges of A: $\leq n + 2k$
number of walls of T: $\leq 2(n + k)$
size of A and T: $O(n + k)$

1st pb. $\Theta(n^2)$ intersection tests

$$[p_0p_1] \cap [p_2p_3] \neq \emptyset$$

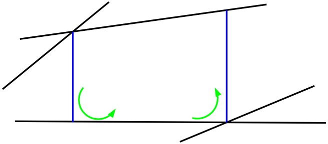
2nd pb. Description of A uses



$$[p_0p_1] \cap [p_2p_3] <_x [p_0p_1] \cap [p_4p_5]$$

comparisons of constructed points

3rd pb. Description of T uses



$$[p_0p_1] \cap [p_2p_3] <_x [p_4p_5] \cap [p_6p_7]$$

$$
\begin{array}{ll}
\texttt{P1}\ : & p_0 <_x p_1 \\
\texttt{P2}\ : & p_0 <_y (p_1 p_2) \\
\texttt{P2}':& [p_0 p_1] \cap [p_2 p_3] \neq \emptyset \\
\texttt{P3}\ : & p_0 <_x [p_1 p_2] \cap [p_3 p_4] \\
\texttt{P4}\ : & [p_0 p_1] \cap [p_2 p_3] <_x [p_0 p_1] \cap [p_4 p_5] \\
\texttt{P5}\ : & [p_0 p_1] \cap [p_2 p_3] <_x [p_4 p_5] \cap [p_6 p_7]
\end{array}
$$

Predicates i, i' are signs of polynomial expressions of degree i in the coordinates of points p_j.

`P1 :` $p_0 <_x p_1$

$$x_0 < x_1$$

degree 1

Arrangement
Predicates

`P1 :` $p_0 <_x p_1$

$$x_0 < x_1$$

degree 1

`P2 :` $p_0 <_y (p_1 p_2)$

orientation

degree 2

Arrangement
Predicates

P1 : $p_0 <_x p_1$

$$x_0 < x_1$$

degree 1

P2 : $p_0 <_y (p_1 p_2)$

orientation

degree 2

P2′: $[p_0 p_1] \cap [p_2 p_3] \neq \emptyset$

compare + 2 $\times$ orientation

degree 2

$$[p_i p_j] \cap [p_k p_l] = p_i + (p_j - p_i)\frac{N}{D}$$

where

$$\begin{aligned} N &= \text{orientation}(p_i, p_k, p_l) \\ D &= \text{orientation}(p_i, p_j, p_k) - \text{orientation}(p_i, p_j, p_l) \end{aligned}$$

$$[p_i p_j] \cap [p_k p_l] = p_i + (p_j - p_i)\frac{N}{D}$$

where

$$\begin{aligned} N &= \text{orientation}(p_i, p_k, p_l) \\ D &= \text{orientation}(p_i, p_j, p_k) - \text{orientation}(p_i, p_j, p_l) \end{aligned}$$

$$\begin{aligned} \text{P3} &: \quad p_0 <_x [p_1 p_2] \cap [p_3 p_4] \\ \text{P4} &: \quad [p_0 p_1] \cap [p_2 p_3] <_x [p_0 p_1] \cap [p_4 p_5] \\ \text{P5} &: \quad [p_0 p_1] \cap [p_2 p_3] <_x [p_4 p_5] \cap [p_6 p_7] \end{aligned}$$

Explicit formulae + some more proofs $\longrightarrow$ degree

1st pb. Compute the pairs of segment that intersect

Naive algorithm $\Theta(n^2)$

- optimal degree 2
- optimal worst-case complexity

1st pb. Compute the pairs of segment that intersect

Naive algorithm $\Theta(n^2)$

- optimal degree 2
- optimal worst-case complexity

Lower bound $\Omega(n \log n + k)$.
There are algorithms

- optimal complexity
- degree 3

2nd pb. Compute the arrangement A

Simple algorithm:
- solve 1st pb
- sort intersection points on each segment

- degree 4
- $O((n+k)\log n)$

Lower bound $\Omega(n \log n + k)$.

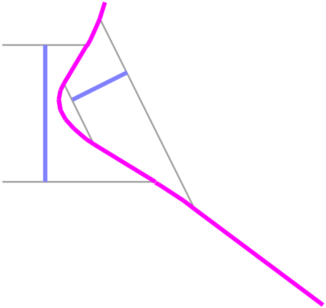
Curved objects

- the world is not linear
  - CAD
  - structural biology
  - ...

Curved objects

- the world is not linear
  - CAD
  - structural biology
  - ...
- curves appear with linear input:

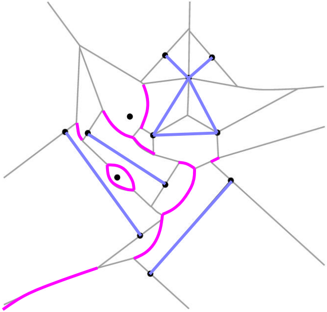
  Voronoi diagrams of line segments
  = subset of arrangement of curves



Curved objects

- the world is not linear
  - CAD
  - structural biology
  - ...
- curves appear with linear input:

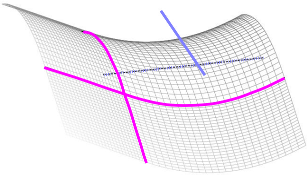
  Voronoi diagrams of line segments
  = subset of arrangement of curves



Curved objects

- the world is not linear
  - CAD
  - structural biology
  - ...
- curves appear with linear input:

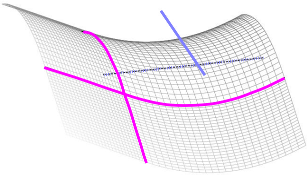
  Voronoi diagrams of line segments
  = subset of arrangement of curves



Curved objects

- the world is not linear
  - CAD
  - structural biology
  - ...
- curves appear with linear input:

  Voronoi diagrams of line segments
  = subset of arrangement of curves

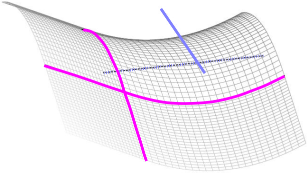


manipulations of curves and surfaces

Curved objects

- the world is not linear
  - CAD
  - structural biology
  - ...
- curves appear with linear input:

  Voronoi diagrams of line segments
  = subset of arrangement of curves



Exact manipulations of curves and surfaces

Arrangements of curves

Combinatorial complexity — well studied

Effective computation ? — recent work

European projects ECG, ACS $\longrightarrow$ CGAL

Problems

- Generalize algorithms

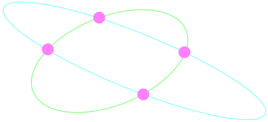
  2 curves intersect more than once,. . .
- Predicates

  algebraic aspects
- Implementation
  - algorithms and data structures
  - predicates

Bézout's theorem:
two curves of degree d, d' intersect in $d.d'$ points

Bézout's theorem:
two curves of degree d, d' intersect in $d.d'$ points

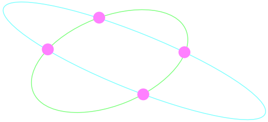
2 conics



degree 4

Bézout's theorem:
two curves of degree d, d' intersect in $d.d'$ points

2 conics



degree 4

2 circles

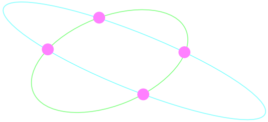
$$\begin{array}{ccccccc} (x-a)^2 & + & (y-b)^2 & - & r^2 & = & 0 \\ (x-a')^2 & + & (y-b')^2 & - & r'^2 & = & 0 \end{array}$$

homogeneization: $x^2 + y^2 + w(...) = 0$
all circles contain $(1, i, 0)$ and $(1, -i, 0)$

Bézout's bound: complex projective space

Bézout's theorem:
two curves of degree d, d' intersect in $d.d'$ points

2 conics



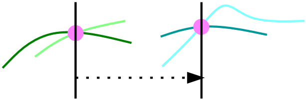
degree 4

2 circles

$$\begin{array}{ccccccccc} (x-a)^2 & + & (y-b)^2 & - & r^2 & = & 0 \\ (x-a')^2 & + & (y-b')^2 & - & r'^2 & = & 0 \end{array}$$

$\Longleftrightarrow$ 1 circle and 1 line (radical axis)
degree 2

Bézout's bound: complex projective space

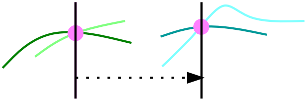


major predicate

points’ coordinates = algebraic numbers

Question:

exact comparison of algebraic numbers



major predicate

points' coordinates = algebraic numbers

Question:

exact comparison of algebraic numbers

(note: Different notions of degree

- above: degree of polynomial expressions
- here: degree of roots)

2 main approaches

- root isolation and comparison of intervals
  when roots are very close or equal
  up to separation bound $\longrightarrow$ very slow

2 main approaches

- root isolation and comparison of intervals
  when roots are very close or equal
  up to separation bound $\longrightarrow$ very slow
- algebraic methods for root comparison
  not sensitive to special cases

Sturm sequences

$P, Q \in \mathbb{K}[X]$ signed remainder sequence of P and Q = sequence $\mathcal{S}(P,Q) : P_0, P_1, \ldots, P_k$

$$
\begin{aligned}
P_0 &= P \\
P_1 &= Q \\
P_2 &= -Rem(P_0, P_1) \\
&\vdots \\
P_k &= -Rem(P_{k-2}, P_{k-1}) \\
P_{k+1} &= -Rem(P_{k-1}, P_k) = 0
\end{aligned}
$$

where
$Rem(A,B) =$ remainder of the Euclidean division of A by B

Sturm sequences

$a, b \in \mathbb{R} \cup \{-\infty, +\infty\}$

$Var(\mathcal{S}; a)$ = number of sign variations in the sequence $P_0(a), P_1(a), \ldots, P_d(a)$

$$Var(\mathcal{S}; a, b) = Var(\mathcal{S}; a) - Var(\mathcal{S}; b)$$

Sturm sequences allow to

- count roots

Sturm sequence of P = $\mathcal{S}(P, P')$

$$Var(\mathcal{S}(P, P'); a, b)$$

is the number of roots of P in the interval $[a, b]$

Sturm sequences allow to

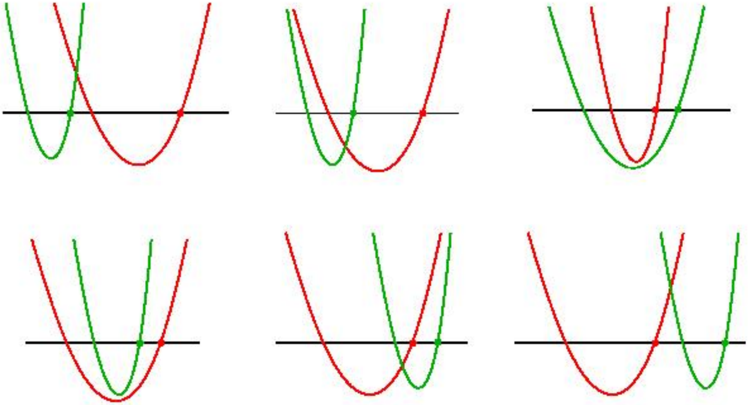
- count roots
- compare roots

- P, Q relative prime,
- P square free,
- $a < b$ non roots of P.

$\mathcal{S} = (P, P'Q, \dots)$ Sturm sequence of $P, P'Q$

$$Var(\mathcal{S}; a, b) = \sum_{P(\rho)=0,\, a<\rho<b} \operatorname{sign}(Q(\rho))$$

Case of degree 2. P Q



comparison reduces to

sign of algebraic expressions !

$\longrightarrow$ Efficient filtered exact computations

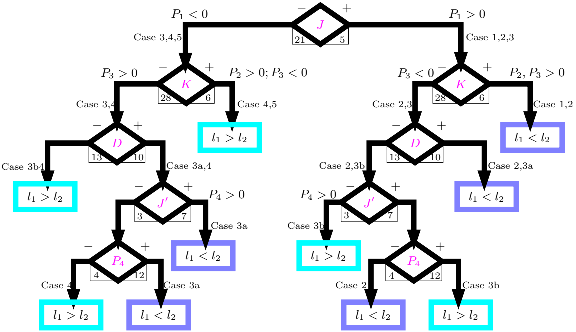
Arrangements of curves

Comparison of algebraic numbers

Small degree:

algebraic expressions can be pre-computed

static Sturm sequences (degree 2)

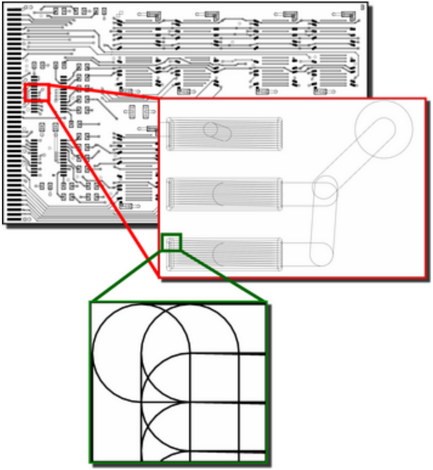


The polynomial expressions have a true geometric meaning

Sturm sequences $\Leftrightarrow$ resultant based methods. . .

CGAL and applications

- generic arrangements
- manipulations of 2d circular arcs



VLSI design

industrial data
89,918 input arcs
495,209 vertices
878,799 edges
383,871 faces

CGAL 3.3: 169 sec
Pentium 4, 2.5 GHz, 1GB
Linux (2.4.20 Kernel)
g++4.0.2

Hot topics

- exact drawing of curves
  = any zoom possible

Hot topics

- exact drawing of curves
  = any zoom possible
- exact topology of curves

Hot topics

- exact drawing of curves
  = any zoom possible
- exact topology of curves
- arrangements of quadrics, spheres
  - surfacic approaches
  - volumic approaches

algebraic issues, data structures. . .

Hot topics

- exact drawing of curves
  = any zoom possible
- exact topology of curves
- arrangements of quadrics, spheres
  - surfacic approaches
  - volumic approaches

  algebraic issues, data structures…
- design of interface geometry/algebra
  geometric/algebraic concepts // C++ concepts

Hot topics

- exact drawing of curves
  = any zoom possible
- exact topology of curves
- arrangements of quadrics, spheres
  - surfacic approaches
  - volumic approaches

  algebraic issues, data structures. . .
- design of interface geometry/algebra
  geometric/algebraic concepts // C++ concepts
- definition of the degree of predicates/algorithm/problem

Where it happens (unordered - non exhaustive)

USA

- NYU (Chee Yap, pioneer of the Exact Geometric Computation, CORE library)
- University of N. Carolina (Dinesh Manocha *et al*, MAPC, ESOLID no degeneracies allowed)

Where it happens (unordered - non exhaustive)

Mostly in Europe

- MPI Saarbrücken (arrangements of 2d cubics, 3d quadrics, EXACUS prototype $\rightarrow$ CGAL)
- Tel-Aviv (generic arrangements of curves, CGAL)
- Athens (algebraic aspects, Voronoi of conics)

Where it happens (unordered - non exhaustive)

in France

- INRIA Rocquencourt/UPMC
  SALSA, real algebraic geometry, RUR,
  software FGb/RS ($\rightarrow$ Maple)
- INRIA Lorraine
  VEGAS, quadrics, Voronoi of 3D lines
- INRIA Sophia Antipolis
  GEOMETRICA + ABS
  arrangements of spheres,
  computations on 2d/3d circular arcs,
  specifications of curved and algebraic operations (with MPI),
  CGAL design and implementation
- collaboration on interface FBb/RS $\leftrightarrow$ CGAL