Delaunay triangulation:

Implementation

Monique Teillaud
Choosing an algorithm

(not only) laziness

Incremental algorithm

fully dynamic

any dimension
Choosing an algorithm

(not only) laziness

Incremental algorithm

fully dynamic
any dimension
Choosing an algorithm

(not only) laziness

Incremental algorithm

fully dynamic
any dimension
Choosing an algorithm

(not only) laziness

Incremental algorithm

fully dynamic
any dimension
Choosing an algorithm

(not only) laziness

Incremental algorithm

fully dynamic

any dimension
Representation

walk: access to
  • vertices of a triangle
  • neighbors of a triangle
in **constant** time
Representation

walk: access to
  • vertices of a triangle
  • neighbors of a triangle
in constant time

combinatorics:
store
  • $d$-simplices
  • vertices

adjacency relations as pointers

geometry
store
  • points in vertices
Representation

walk: access to
- vertices of a triangle
- neighbors of a triangle
in constant time

combinatorics:
store
- $d$-simplices
- vertices

geometry

adjacency relations as pointers

what about the infinite region? unbounded size...
Representation

add a bounding box?

requires to know points in advance
Representation

add a bounding box?

requires to know points in advance
creates ugly triangles
Representation

compactification of $\mathbb{R}^d$  
$\implies$ triangulation of the sphere $S^d$

add a vertex at infinity
Representation

combinatorics
all $d$-faces are simplices
constant description
constant-time access...

geometry
infinite simplex = half-space
no point in the infinite vertex
what if all points are collinear?
what if all points are collinear?

$dD$ triangulation, $d \geq 2$

"incomplete" simplices

triangulation of $S^1$

what if all points are collinear?
Representation

what if all points are collinear?
what if a non-collinear point comes in?
Representation

what if all points are collinear?

what if a non-collinear point comes in?

triangulation of $S^1$

$\rightarrow$ triangulation of $S^2$
what if all points are collinear?
what if a non-collinear point comes in?
what if a non-coplanar point comes in?
Arithmetic computations

Input

Predicates

\(< 0 \quad = 0 \quad > 0\)

Constructions

Combinatorial Structure

Geometric embedding
Arithmetic computations
Arithmetic computations

Combinatorial structure

only predicates
Arithmetic computations

Geometric embedding

constructions

same underlying combinatorial structure
Arithmetic computations

inexact evaluation of predicates

NOT just an imprecision in the result

\[ \text{in\_disk}_1(p) = \text{true} \]

\[ \text{in\_disk}_2(p) = \text{false} \]
Arithmetic computations

inexact evaluation of predicates

NOT just an imprecision in the result

\[ \text{in\_disk}_1(p) = \text{true} \]
\[ \text{in\_disk}_2(p) = \text{false} \]

inconsistencies!

algorithms fail
Is $s$ inside or outside the disk?
Arithmetic computations

Is \( s \) inside or outside the disk?

Circle \( C \) through \( p, q, r \)

Unknowns \( c, \rho \)

Solve \( \rightarrow \)

- Center \( c \)
- Radius \( \rho \)
Arithmetic computations

Is $s$ inside or outside the disk?

Circle $C$ through $p, q, r$

Unknowns $c, \rho$

Solve $\rightarrow$
- Center $c$
- Radius $\rho$

Bad idea... reals do not exist!

Rounding errors $\mapsto p, q, r \notin C(c, \rho)$

"Random" result for $s$
Arithmetic computations

Is $s$ inside or outside the disk?

Predicate $\text{in\_disk}(p, q, r, s)$  

$$\begin{vmatrix}
1 & 1 & 1 & 1 \\
p_x & qx & r_x & s_x \\
p_y & qy & r_y & s_y \\
p_x^2 + p_y^2 & q_x^2 + q_y^2 & r_x^2 + r_y^2 & s_x^2 + s_y^2
\end{vmatrix}$$
Arithmetic computations

A simpler predicate
Is \( r \) on the left or right side?

\[
\text{orient}(p, q, r)
\]

\[
\text{sign} \begin{vmatrix}
1 & 1 & 1 \\
px & qx & rx \\
py & qy & ry \\
\end{vmatrix}
\]
Arithmetic computations

**double numbers are not reals**

53 binary digits

\[ p = (0.5 + x.u, 0.5 + y.u) \]
\[ 0 \leq x, y < 256, \]
\[ u = 2^{-53} \]

\[ q = (12, 12) \]
\[ r = (24, 24) \]

orient\((p, q, r)\)

- **> 0**
- **= 0**
- **< 0**

\((x, y) = (256, 256)\)

\((x, y) = (0, 0)\)

fast, but **wrong**
Arithmetic computations

a solution:

rely on an exact arithmetic package (multiprecision, etc)

powerful, but slow
Arithmetic computations

Exact Geometric Computing paradigm

= exact predicates, ≠ exact arithmetics

Filtering

Approximate evaluation $P^a(x)$
+ Error $\varepsilon$

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

? y n

sign $(P(x)) = \text{sign} (P^a(x))$

Exact computation

easy cases are more frequent
$\implies$ cost $\simeq$ cost of approximate (double) computation
Arithmetic computations

Dynamic filtering interval arithmetic

error on $+,-,\ast,\div,\sqrt{}$ known (IEEE 754)

$$[a,a] + [b,b] = [a+a, a+b]$$

and propagate...
Arithmetic computations

Dynamic filtering interval arithmetic

error on $+,-,*,/,$ $\sqrt{}$ known (IEEE 754)

$[a, \overline{a}] + [b, \overline{b}] = [a + a, \overline{a} + \overline{b}]$

and propagate...

$0 \notin [\text{result}, \overline{\text{result}}]$
Choosing an algorithm

Degree of predicates & number of operations

→ constant in $O()$

→ size of errors

→ length of integers for exact arithmetic
Choosing an algorithm

Degree of predicates & number of operations

$\rightarrow$ constant in $O()$

$\rightarrow$ size of errors

$\rightarrow$ length of integers for exact arithmetic

Incremental algorithm

only uses intrinsic predicates

orient, in_disk

any algorithm computing Delaunay triangulation is able to answer them

Sweep

uses ad hoc higher degree predicates
Choosing an algorithm

Degree of predicates & number of operations

\[ \rightarrow \text{constant in } O() \]

\[ \rightarrow \text{size of errors} \]

\[ \rightarrow \text{length of integers for exact arithmetic} \]

Incremental algorithm

only uses intrinsic predicates

orient, in_disk

any algorithm computing Delaunay triangulation is able to answer them

Sweep

uses ad hoc higher degree predicates

laziness is not the only criterion
Degeneracies
Degeneracies

Exact computation

\[ \text{in_disk}(\cdot, \cdot, \cdot, p) = 0 \]

what if \( p \) lies on a circle?

yes, it does happen!

input data are rounded
Degeneracies

Delaunay complex

non-simplicial faces
Degeneracies

Delaunay complex
non-simplicial faces

non-constant storage and access

users want triangles
Degeneracies

“the” Delaunay triangulation is not unique
Degeneracies

Simulating the absence of degeneracies
Degeneracies

Simulating the absence of degeneracies

as if \( p \) outside
Degeneracies

Simulating the absence of degeneracies

as if $p$ inside
Degeneracies
Degeneracies

decisions must be made in a consistent way
Degeneracies

Symbolic perturbation

Input data $\mapsto$ data depending on a symbolic parameter $\varepsilon$

- $\varepsilon = 0$: (maybe) degenerate problem
- $\varepsilon \neq 0$: non-degenerate problem $\mapsto$ Result($\varepsilon$)

Final result $= \lim_{\varepsilon \to 0^+} \text{Result}(\varepsilon)$
Degeneracies

SoS: simulation of simplicity

Input: $n$ points $p_i = (x_i, y_i), i = 1, \ldots, n$

$\forall i, (x_i, y_i) \mapsto (x_i, y_i) + \varepsilon^{2^i}(i, i^2)$
Degeneracies

SoS: simulation of simplicity

Input: \( n \) points \( p_i = (x_i, y_i), i = 1, \ldots, n \)

\( \forall i, (x_i, y_i) \mapsto (x_i, y_i) + \varepsilon^{2i}(i, i^2) \)

\[
\text{orient}(O, p_i, p_i) = \text{sign} \begin{vmatrix} x_i & x_j \\ y_i & y_j \end{vmatrix}
\]

\[
\begin{vmatrix} x_3 & x_1 \\ y_3 & y_1 \end{vmatrix} \mapsto \begin{vmatrix} x_3 + 3\varepsilon^8 & x_1 + \varepsilon^2 \\ y_3 + 9\varepsilon^8 & y_1 + \varepsilon^2 \end{vmatrix} =
\]

\[
\begin{vmatrix} x_3 & x_1 \\ y_3 & y_1 \end{vmatrix} + \varepsilon^2 \begin{vmatrix} x_3 & 1 \\ y_3 & 1 \end{vmatrix} + \varepsilon^8 \begin{vmatrix} 3 & x_1 \\ 9 & y_1 \end{vmatrix} + \varepsilon^{10} \begin{vmatrix} 3 & 1 \\ 9 & 1 \end{vmatrix}
\]
Degeneracies

SoS: simulation of simplicity

Input: $n$ points $p_i = (x_i, y_i), i = 1, \ldots, n$

$\forall i, (x_i, y_i) \mapsto (x_i, y_i) + \epsilon^{2i}(i, i^2)$

$$orient(O, p_i, p_i) = \text{sign} \left| \begin{array}{cc} x_i & x_j \\ y_i & y_j \end{array} \right|$$

$$\left| \begin{array}{cc} x_3 & x_1 \\ y_3 & y_1 \end{array} \right| \mapsto \left| \begin{array}{cc} x_3 + 3\epsilon^8 & x_1 + \epsilon^2 \\ y_3 + 9\epsilon^8 & y_1 + \epsilon^2 \end{array} \right| =$$

$$\left| \begin{array}{cc} x_3 & x_1 \\ y_3 & y_1 \end{array} \right| + \epsilon^2 \left| \begin{array}{cc} x_3 & 1 \\ y_3 & 1 \end{array} \right| + \epsilon^8 \left| \begin{array}{cc} 3 & x_1 \\ 9 & y_1 \end{array} \right| + \epsilon^{10} \left| \begin{array}{cc} 3 & 1 \\ 9 & 1 \end{array} \right|$$

sign = sign of first non-null coefficient
Degeneracies

SoS: simulation of simplicity

Input: $n$ points $p_i = (x_i, y_i), i = 1, \ldots, n$

$\forall i, (x_i, y_i) \mapsto (x_i, y_i) + \varepsilon^{2i}(i, i^2)$

$\text{orient}(O, p_i, p_i) = \text{sign} \begin{vmatrix} x_i & x_j \\ y_i & y_j \end{vmatrix}$

$\longrightarrow$ always $> 0$ or $< 0$

same for in\_disk
Degeneracies

SoS: simulation of simplicity

Input: $n$ points $p_i = (x_i, y_i), i = 1, \ldots, n$

$\forall i, (x_i, y_i) \mapsto (x_i, y_i) + \varepsilon^{2i}(i, i^2)$

orient($O, p_i, p_i$) = sign \[
\begin{vmatrix}
  x_i & x_j \\
  y_i & y_j \\
\end{vmatrix}
\]

$\rightarrow$ always $> 0$ or $< 0$

same for in_disk

may create FLAT simplices
Degeneracies

Perturbing points in $d + 1^{th}$ dimension

$\pi_i = (x_i, y_i, t_i = x_i^2 + y_i^2)$

$p_i = (x_i, y_i)$
Degeneracies

Perturbing points in $d + 1^{th}$ dimension

$$\pi_i = (x_i, y_i, t_i = x_i^2 + y_i^2)$$

$$p_i = (x_i, y_i)$$

$$\text{in}_\text{disk}(p_i, p_j, p_k, p_l) = \frac{\mathcal{D}(p_i, p_j, p_k, p_l)}{\text{orient}(p_i, p_j, p_k)}$$

$$\mathcal{D}(p_i, p_j, p_k, p_l) = \text{orient}(\pi_i, \pi_j, \pi_k, \pi_l)$$
Degeneracies

Perturbing points in \(d + 1\)th dimension

\[
in_{\text{disk}}(p_i, p_j, p_k, p_l) = \frac{D(p_i, p_j, p_k, p_l)}{\text{orient}(p_i, p_j, p_k)}
\]

\[
D(p_i, p_j, p_k, p_l) = \text{orient}(\pi_i, \pi_j, \pi_k, \pi_l) \mapsto \text{orient}(\pi_i^\varepsilon, \pi_j^\varepsilon, \pi_k^\varepsilon, \pi_l^\varepsilon)
\]

\[
\pi_i = (x_i, y_i, t_i = x_i^2 + y_i^2) \mapsto \pi_i^\varepsilon = (x_i, y_i, t_i + \varepsilon^{n-i})
\]

\[
p_i = (x_i, y_i)
\]
Degeneracies

Perturbing points in $d + 1^{\text{th}}$ dimension

$$\text{orient}(\pi^\varepsilon_i, \pi^\varepsilon_j, \pi^\varepsilon_k, \pi^\varepsilon_l) = \begin{vmatrix}
1 & 1 & 1 & 1 \\
x_i & x_j & x_k & x_l \\
y_i & y_j & y_k & y_l \\
z_i & z_j & z_k & z_l \\
t_i + \varepsilon^{n-i} & t_j + \varepsilon^{n-j} & t_k + \varepsilon^{n-k} & t_l + \varepsilon^{n-l}
\end{vmatrix}$$

$$= D(p_i, p_j, p_k, p_l) - \text{orient}(p_i, p_j, p_k)\varepsilon^{n-l} + \text{orient}(p_i, p_j, p_l)\varepsilon^{n-k} - \text{orient}(p_i, p_k, p_l)\varepsilon^{n-j} + \text{orient}(p_j, p_k, p_l)\varepsilon^{n-i}$$

4 cocircular points $\rightarrow$ non-null polynomial in $\varepsilon$

point with highest index
in the disk of the other 3
Degeneracies

Perturbing points in \( d + 1 \text{th} \) dimension

orientation predicate not perturbed

\[ \implies \text{NO flat simplex created} \]

global indexing = lexicographic order

\[ \implies \text{Delaunay triangulation uniquely defined} \]

easy to implement
Computational Geometry Algorithms Library

www.cgal.org

open source
distributed under GPL
commercial licences distributed by GeometryFactory
Computational Geometry Algorithms Library

CGAL

Computational Geometry Algorithms Library

www.cgal.org

2D, 3D, $dD$ [weighted] Delaunay triangulations

2D $\approx$ 10 million points / second
3D $\approx$ 1 million points / second

(on a standard laptop)

fully dynamic

fully robust
2D, 3D periodic [weighted] Delaunay triangulations (flat torus)
Computational Geometry Algorithms Library

www.cgal.org

soon (?)

hyperbolic Delaunay triangulations
Delaunay triangulations on the sphere
used by astrophysicists, biologists,
Computational Geometry Algorithms Library

www.cgal.org

used by astrophysicists, biologists, mathematician(s?) ...
Take home (?)

There is a long way from the algorithm to the software

Needed

clean mathematical models

good algorithms

knowledge of computers

union makes strength