Reliable and Efficient Geometric Computation

via Controlled Perturbation

Kurt Mehlhorn
Max-Planck-Institut für Informatik

slides and papers are available at my home page
Overview

- Controlled Perturbation
  - The Principle
  - Applicability and Limits
  - Practicability
  - Open Problems

- Sources
  - KM, R. Osbild, M. Sagralof: Reliable and Efficient Computational Geometry via Controlled Perturbation (Extended Abstract), ICALP 2006
  - and the papers cited therein, in particular, Dany Halperin’s papers
Approaches to Reliable and Efficient Geometric Computing

• Solutions for single algorithms.

• The Exact Geometric Computation Paradigm (ECG)
  • implement a Real-RAM to the extent needed in computational geometry
  • the challenge is an efficient realization
  • not the subject of this lecture

• Approximation
  • compute the correct result for a slightly perturbed input
  • Controlled Perturbation
    • actively choose the perturbed input
    • initiated by Danny Halperin and co-workers
    • refined and generalized by us
    • message of the day: controlled perturbation is a general method applicable to a large class of geometric algorithms
The Orientation Predicate

three points $p$, $q$, and $r$ in the plane either lie

- on a common line or form a left or right turn $\text{orient}(p, q, r) = 0$, $+1$, $-1$

- analytically

$$\text{orient}(p, q, r) = \text{sign}(\det \begin{bmatrix} 1 & p_x & p_y \\ 1 & q_x & q_y \\ 1 & r_x & r_y \end{bmatrix})$$

$$= \text{sign}
\begin{pmatrix}
(q_x - p_x)(r_y - p_y) - (q_y - p_y)(r_x - p_x)
\end{pmatrix}.$$

- det is twice the signed area of the triangle $(p, q, r)$
- $\text{float}_\text{orient}(p, q, r)$ is result of evaluating $\text{orient}(p, q, r)$ in floating point arithmetic
Geometry of Float-Orient

\[ p = (0.5, 0.5), \ q = (12, 12) \text{ and } r = (24, 24) \]

picture shows

\[
\text{float}_\text{orient}((p_x + xu, p_y + yu), q, r)
\]

for \( 0 \leq x, y \leq 255 \), where \( u = 2^{-53} \).

the line \( \ell(q, r) \) is shown in black

near the line many points are mis-classified

in L. Kettner, KM, S. Pion, S. Schirra, C. Yap: Classroom Examples of Robustness Problems in Geometric Computations, ESA 2004, LNCS 3221, 702–713, we show how this leads to global failures in floating point implementations of geometric algorithms
Geometry of Float-Orient

- picture shows

$$\text{float}_\text{orient}((p_x + xu, p_y + yu), q, r)$$

for $0 \leq x, y \leq 255$, where $u = 2^{-53}$.

- the line $\ell(q, r)$ is shown in black

- near the line many points are mis-classified

- outside a narrow strip around the curve of degeneracy, points are classified correctly !!!

- why is this the case and how narrow is narrow?
- true for all geometric predicates?
- if true, can we exploit to design reliable algorithms
The Idea Behind Controlled Perturbation

- $\text{float\_orient}(p, q, r)$ seems to work fine, if $p, q, \text{and } r$ are not nearly collinear

- A similar statement holds for all geometric predicates
  - geometric predicate = $\text{sign} f(p_1, \ldots, p_k)$
  - fix $p_1$ to $p_{k-1}$ and consider $g(p_k) := f(p_1, \ldots, p_k)$
  - either identically zero
  - or defines a curve $C = C(p_1, \ldots, p_{k-1})$
  - if $p_k$ is sufficiently far from this curve, floating point evaluation of $f$ gives the right sign

- How can we guarantee
  - that no three points are nearly collinear
  - that $p_k$ does not lie near the curve $C, \ldots$?

- Perturb the points by random amounts and this should do it
A Visualization of Controlled Perturbation

Possible Perturbations

Forbidden Areas
Basics

- our program operates on points $q_1$ to $q_n$

  to perturb a point $q_i$:
  - move it to random point $p_i$ in the disk $B_\delta(q_i)$ of radius $\delta$ centered at $q_i$

- programs branch on the sign of expressions
- we use floating point arithmetic with mantissa length $p$
- the maximum error in evaluating an expression $E$ is $M_E$
  - $M_E = \text{something} \cdot 2^{-p}$
- if $|E| > M_E$, it is safe to evaluate $E$ with floating point arithmetic and to branch on the sign of the result
- Assumption: we have a geometric program that works for all non-degenerate inputs (if executed with exact real arithmetic)
Converting a Program to Controlled Perturbation

• guard every predicate evaluation, i.e.,
  
  replace branch on sign of $E$ by

  if ($|E| \leq \text{max error in evaluation of } E$) stop with exception;
  branch on sign of $E$

• and then run the following master program
  
  • initialize $\delta$ and $p$ to convenient values
  • loop
    • perturb input
    • run the guarded algorithm with floating point precision $p$
    • if the program fails, double $p$ and rerun

• observe that program needs to be changed only slightly, namely guards for predicates and master loop

• guards can be avoided by use of interval arithmetic
Converting a Program to Controlled Perturbation

• guard every predicate evaluation, i.e.,

  replace branch on sign of $E$ by

  if ($|E| \leq \text{max error in evaluation of } E$) stop with exception;
  branch on sign of $E$

• and then run the following master program
  • initialize $\delta$ and $p$ to convenient values
  • loop
    • perturb input
    • run the guarded algorithm with floating point precision $p$
    • if the program fails, double $p$ and rerun

Theorem: For a large class of geometric programs: modified program terminates and returns the exact result for the perturbed input.

Moreover, we can quantify the relation between $p$ and $\delta$ at termination
Why does \textit{float\_orient} work outside a narrow strip?

- \( orient(p, q, r) = \text{sign}((q_x - p_x)(r_y - p_y) - (q_y - p_y)(r_x - p_x)) = \text{sign}(E) \)

- \( E = 2 \cdot \text{signed area } \Delta \text{ of the triangle } (p, q, r) \)

- \(|\Delta| = (1/2) \cdot \text{dist}(q, r) \cdot \text{dist}(\ell(q, r), p) \)

- \( E = \text{dist}(q, r) \cdot \text{dist}(\ell(q, r), p) \)
Why does \texttt{float\_orient} work outside a narrow strip?

- \(\text{orient}(p, q, r) = \text{sign}((q_x - p_x)(r_y - p_y) - (q_y - p_y)(r_x - p_x)) = \text{sign}(E)\)

- \(E = 2 \cdot \text{signed area } \Delta \text{ of the triangle } (p, q, r)\)

- \(|\Delta| = (1/2) \text{dist}(q, r) \cdot \text{dist}(\ell(q, r), p)\)

- \(E = \text{dist}(q, r) \cdot \text{dist}(\ell(q, r), p)\)

- \(E\) measures the distance of \(p\) from the curve of degeneracy and hence round-off cannot invalidate the result, if \(p\) is sufficiently far from the curve.
How Narrow is Narrow?

- \( \text{orient}(p, q, r) = \text{sign}((q_x - p_x)(r_y - p_y) - (q_y - p_y)(r_x - p_x)) = \text{sign}(E) \)
- \( E = \text{dist}(q, r) \cdot \text{dist}(\ell(q, r), p) \)
- If coordinates are bounded by \( M \), maximal error in evaluating \( E \) with floating point arithmetic with mantissa length \( p \) is \( 28 \cdot M^2 \cdot 2^{-p} \)
  - Deal with numbers as large as \( 4M^2 \)
  - Error in a single operation is at most \( 4M^22^{-p} \)
  - 7 accounts for the number of operations
How Narrow is Narrow?

- \( \text{orient}(p, q, r) = \text{sign}((q_x - p_x)(r_y - p_y) - (q_y - p_y)(r_x - p_x)) = \text{sign}(E) \)

- \( E = \text{dist}(q, r) \cdot \text{dist}(\ell(q, r), p) \)

- if coordinates are bounded by \( M \), maximal error in evaluating \( E \) with floating point arithmetic with mantissa length \( p \) is \( 28 \cdot M^2 \cdot 2^{-p} \)

- if \( \text{dist}(q, r) \cdot \text{dist}(\ell(q, r), p) > 28 \cdot M^2 \cdot 2^{-p} \), \( \text{float}_\text{orient} \) gives the correct result

- Punch Line: if
  \[
  \text{dist}(\ell((q, r), p)) \geq 28 \cdot M^2 \cdot 2^{-p} / \text{dist}(q, r),
  \]
  \( \text{float}_\text{orient}(p, q, r) \) gives the correct result.
Summary: Orientation Predicate

- Punch Line: if
  \[ \text{dist}(\ell((q,r),p)) \geq 28 \cdot M^2 \cdot 2^{-p}/\text{dist}(q,r), \]
  \[ \text{float}_\text{orient}(p,q,r) \text{ gives the correct result.} \]

- forbidden region for \( p \) = a strip of half-width \( 28 \cdot M^2 \cdot 2^{-p}/\text{dist}(q,r) \) about \( \ell(q,r) \)

Observe: width of forbidden region depends on \( \text{dist}(q,r) \)

- if \( p \) lies outside the forbidden region, the evaluation of \( \text{orient}(p,q,r) \) is floating-point safe (f-safe)
Side of Oriented Circle

• can we analyse other predicates in the same way?

\[ \text{side}_\text{of}_\text{circle}(p, q, r, s) = +1, 0, -1 \text{ if } s \text{ lies left of, on, right of oriented circle } C(p, q, r) \]

• analytically: \( \text{side}_\text{of}_\text{circle}(p, q, r, s) = \text{sign} \left| \begin{array}{ccc}
1 & x_1 & y_1 \\
1 & x_2 & y_2 \\
1 & x_3 & y_3 \\
1 & x & y
\end{array} \right| \)

\[ det = 2 \cdot \Delta \cdot (R + \text{dist}(C, s)) \cdot \text{dist}(C, s) \text{ and hence} \]

\[ |det| \geq 2 \cdot \Delta \cdot R \cdot \text{dist}(C, s) \]

• max error in f-evaluation = \( 40 \cdot M^4 \cdot 2^{-p} \)

• f-eval is correct if \( s \) lies outside an annulus of half-width \( 40 \cdot M^4 \cdot 2^{-p} / (2 \cdot \Delta \cdot R) \)
Controlled Perturbation I

- consider algorithms using only the orientation predicate
- input points $q_1, \ldots, q_n$: **Goal:** perturb into $p_1, \ldots, p_n$ such that all evaluations for the perturbed points are $f$-safe.
Controlled Perturbation I

- consider algorithms using only the orientation predicate
- input points \( q_1, \ldots, q_n \): **Goal**: perturb into \( p_1, \ldots, p_n \) such that all evaluations for the perturbed points are f-safe.
- assume \( p_1 \) to \( p_{n-1} \) are already determined:
  - choose \( p_n \) randomly in a circle of radius \( \delta \) about \( q_n \)
  - want: whp \( p_n \) lies outside all strips of half-width
    \[ 28 \cdot M^2 \cdot 2^{-p} / \text{dist}(p_i, p_j) \text{ about } \ell(p_i, p_j) \text{ for } 1 \leq i < j \leq n - 1 \]
Controlled Perturbation I

- consider algorithms using only the orientation predicate
- input points \( q_1, \ldots, q_n \):  
  \textbf{Goal:} perturb into \( p_1, \ldots, p_n \) such that all evaluations for the perturbed points are f-safe.
- assume \( p_1 \) to \( p_{n-1} \) are already determined:
  - choose \( p_n \) randomly in a circle of radius \( \delta \) about \( q_n \)
  - want: whp \( p_n \) lies outside all strips of half-width \( 28 \cdot M^2 \cdot 2^{-p} / \text{dist}(p_i, p_j) \) about \( \ell(p_i, p_j) \) for \( 1 \leq i < j \leq n - 1 \)
- whp = (fails with prob \( \leq 1/(4n) \))
- prob, choice of any \( p_i \) fails is \( \leq 1/4 \)
- with prob \( \geq 3/4 \) perturbed points are f-safe
- need that strips cover at most fraction \( 1/(4n) \) of ball \( B_\delta(q_n) \)
Controlled Perturbation II

- assume $p_1$ to $p_{n-1}$ are already determined:
  - choose $p_n$ in a circle of radius $\delta$ about $q_n$
  - want: whp $p_n$ lies outside all strips of half-width $28 \cdot M^2 \cdot 2^{-p}/\text{dist}(p_i, p_j)$ about $\ell(p_i, p_j)$ for $1 \leq i < j \leq n - 1$
  - need that strips cover at most fraction $1/(4n)$ of ball $B_\delta(q_n)$
Controlled Perturbation II

• assume $p_1$ to $p_{n-1}$ are already determined:
  • choose $p_n$ in a circle of radius $\delta$ about $q_n$
  • want: whp $p_n$ lies outside all strips of half-width
    \[28 \cdot M^2 \cdot 2^{-p} / \text{dist}(p_i, p_j)\]
    about $\ell(p_i, p_j)$ for $1 \leq i < j \leq n - 1$
  • need that strips cover at most fraction $1/(4n)$ of ball $B_{\delta}(q_n)$
• Problem: no fixed $\delta$ will work, since strips can be arbitrarily wide
Controlled Perturbation II

- assume $p_1$ to $p_{n-1}$ are already determined:
  - choose $p_n$ in a circle of radius $\delta$ about $q_n$
  - want: whp $p_n$ lies outside all strips of half-width $28 \cdot M^2 \cdot 2^{-p} / \text{dist}(p_i, p_j)$ about $\ell(p_i, p_j)$ for $1 \leq i < j \leq n - 1$
  - need that strips cover at most fraction $1 / (4n)$ of ball $B_\delta(q_n)$
- Problem: no fixed $\delta$ will work, since strips can be arbitrarily wide
- IDEA: also guarantee $\text{dist}(p_i, p_j) > \gamma$ for some $\gamma$

then forbidden region =

- $n$ disks of radius $\gamma$ and
- $n^2$ strips of width at most $28 \cdot M^2 \cdot 2^{-p} / \gamma$
- size of forbidden region $\leq n \cdot \pi \cdot \gamma^2 + n^2 \cdot (28 \cdot M^2 \cdot 2^{-p} / \gamma) \cdot 2 \cdot \delta$
Controlled Perturbation II

• assume \( p_1 \) to \( p_{n-1} \) are already determined:
  • choose \( p_n \) in a circle of radius \( \delta \) about \( q_n \)
  • want: whp \( p_n \) lies outside all strips of half-width
    \( 28 \cdot M^2 \cdot 2^{-p} / \text{dist}(p_i, p_j) \) about \( \ell(p_i, p_j) \) for \( 1 \leq i < j \leq n - 1 \)
  • need that strips cover at most fraction \( 1/(4n) \) of ball \( B_\delta(q_n) \)
  • IDEA: also guarantee \( \text{dist}(p_i, p_j) > \gamma \) for some \( \gamma \)
  • size of forbidden region \( \leq n \cdot \pi \cdot \gamma^2 + n^2 \cdot (28 \cdot M^2 \cdot 2^{-p} / \gamma) \cdot 2 \cdot \delta \)
  • want: size of FR \( \leq \pi \cdot \delta^2 / (4n) \)
Controlled Perturbation II

• assume $p_1$ to $p_{n-1}$ are already determined:
  • choose $p_n$ in a circle of radius $\delta$ about $q_n$
  • want: whp $p_n$ lies outside all strips of half-width $28 \cdot M^2 \cdot 2^{-p} / \text{dist}(p_i, p_j)$ about $\ell(p_i, p_j)$ for $1 \leq i < j \leq n - 1$
  • need that strips cover at most fraction $1/(4n)$ of ball $B_\delta(q_n)$
• IDEA: also guarantee $\text{dist}(p_i, p_j) > \gamma$ for some $\gamma$
• size of forbidden region $\leq n \cdot \pi \cdot \gamma^2 + n^2 \cdot (28 \cdot M^2 \cdot 2^{-p} / \gamma) \cdot 2 \cdot \delta$
• want: size of FR $\leq \pi \cdot \delta^2 / (4n)$
• optimal choice of $\gamma$ and simple calculations

• any $p \geq 2 \log(M / \delta) + 4 \log n + 9$ will do
• $M = 1000$, $\delta = 0.001$, $n = 1000$, $p \geq 2 \cdot 20 + 4 \cdot 10 + 9 = 89$, i.e., 89 bits of precision suffice
Summary for Orientation Test

- input: $n$ points in the plane with coordinates bounded by $M$
- alg uses only orientation test
- use floating point arithmetic with mantissa length $p$
- move each point to a random point in its $\delta$-ball
- if

$$p \geq 2\log(M/\delta) + 4\log n + 9$$

then with prob at least $3/4$:

all evaluations of the orientation predicate are f-safe and algorithm will terminate with correct result for perturbed inputs
Generalization to All (??) Geometric Predicates

<table>
<thead>
<tr>
<th>general predicate ( P(x_1, \ldots, x_k) = \text{sign} f(x_1, \ldots, x_k) )</th>
<th>orientation ( \text{orient}(p, q, r) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 ) to ( x_k ) points (in the plane)</td>
<td>( q, r ) fixed, ( p ) variable</td>
</tr>
<tr>
<td>( x = (x_1, \ldots, x_{k-1}) ) fixed, ( x = x_k ) variable</td>
<td>( C = { p : \text{orient}(p, q, r) = 0 } )</td>
</tr>
<tr>
<td>( C_x = { x : f(x, x) = 0 } ), curve of degeneracy</td>
<td>plane or ( \ell(q, r) )</td>
</tr>
<tr>
<td>( C_x ) is either the entire plane or a curve</td>
<td>|</td>
</tr>
</tbody>
</table>

Approach: relate \( f(x, x) \) to the distance of \( x \) from \( C_x \), i.e., to the distance of \( x \) from the curve of degeneracy. Forbidden region becomes tubular neighborhood of this curve. Width of region depends on geometry of \( x \). Recursively guarantee that geometry of \( x \) is nice in the case of the orientation predicate: that any two points have a certain minimal distance.
Summary

- controlled perturbation works for a large class of geometric algorithms:
  - predicates of bounded arity
  - decision trees of depth depending only on number of points in input, but not on actual coordinates

- algs in the class: Delaunay, Voronoi, Arrangements, ....

- computes exact result for a perturbed input

- gives quantitative relation between precision $p$ of floating point system and amount of perturbation $\delta$

- experiments are promising: used successfully for arrangements of spheres and cycles and Delaunay diagram computations
Open Problems

• good versus bad formulas for the same predicate
• treat all predicates required for Voronoi diagrams of line segments. Does it lead to an implementation competitive with Held’s VRONI?
• extension to Descartes algorithm and the algorithms in EXACUS
• can we turn the general scheme into a program transformer, a Controlled-Perturbation-CGAL or a Controlled-Perturbation-EXACUS?

Thanks for your Attention