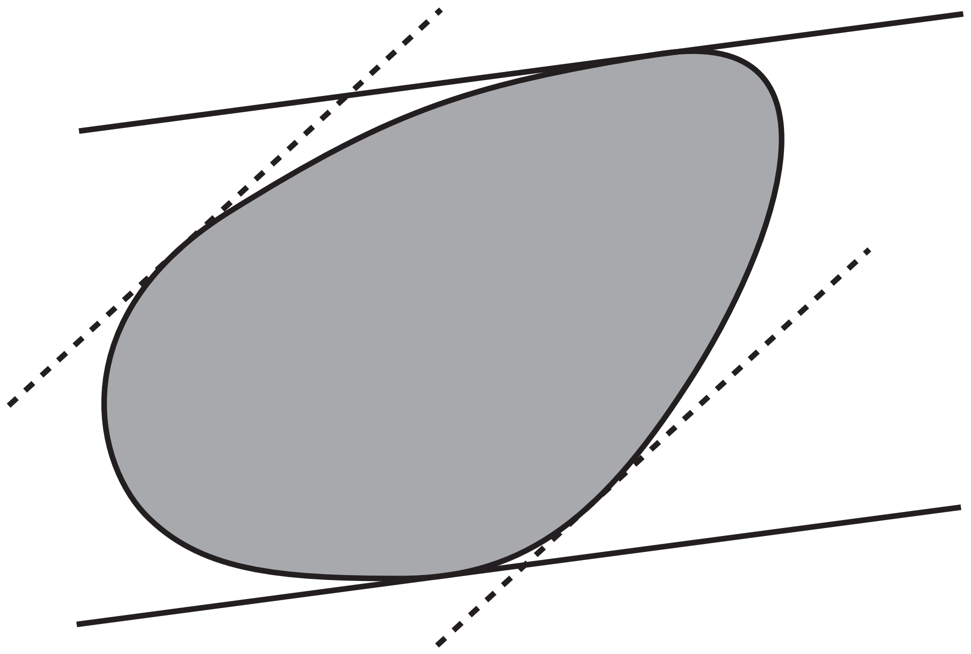
Incremental and Decremental Maintenance of Planar Width

David Eppstein
Dept. Information & Computer Science
Univ. of California, Irvine

Width & Diameter

Any convex body has two tangents
for each possible slope



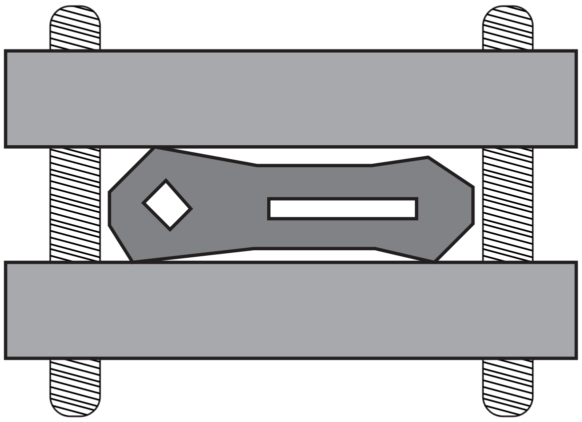
Map slope $\rightarrow$ distance between tangents

Width = min distance
Diameter = max distance

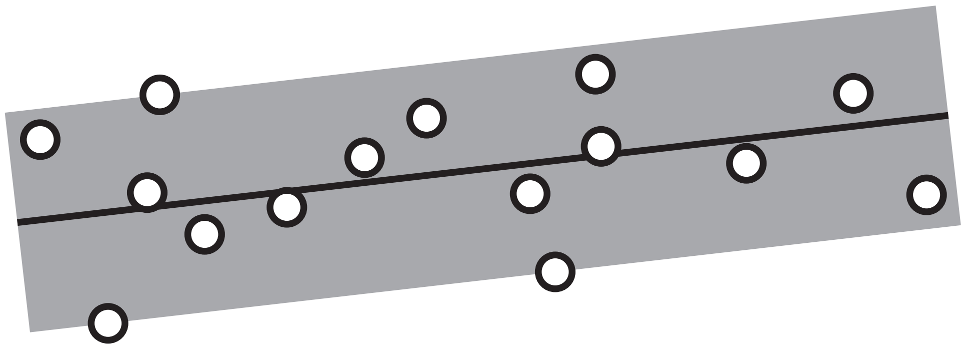
If input is nonconvex, use its convex hull

Why Width?

Stable grip for parallel jaw gripper
(for parts orientation or fixturing)



Statistics: L_∞ regression
(minimize max distance from data to estimator)



Old Results

Static

$O(n \log n)$ [Preparata & Shamos 1985]

Dynamic

Average case for random update sequence
$O(\log n)$ [Eppstein, 1996]

Constant factor approximation
$O(\log^2 n)$ [Janardan; Rote, Schwarz, Snoeyink 1993]

Offline, decision problem only
$O(\log^3 n)$ [Agarwal and Sharir 1991]

Diameter (but not width)
$O(n^\epsilon)$ [Agarwal, Eppstein, Matoušek 1995]

This paper: partially extend diameter result to width

New Results

Width of a dynamic point set
in time $O(kn^{\epsilon})$ per insertion or deletion
where $k =$ number of changes to convex hull

For insertions only (incremental)
or deletions only (decremental)
total hull change $= O(n)$
so amortized time/update $= O(n^{\epsilon})$

Main Idea

Maintain set of convex hull features
(edges and vertices)

[Overmars and van Leeuwen 1981]

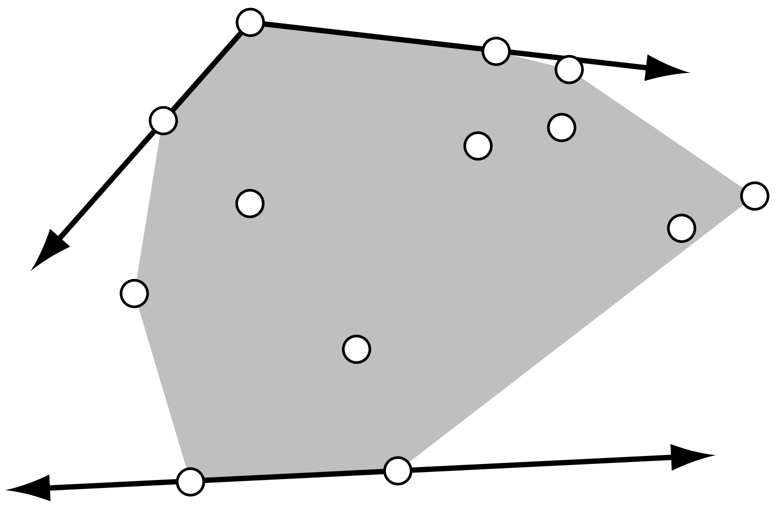
Define distance between features
such that closest pair = width

Build data structures for nearest neighbor queries

Convert nearest neighbors $\Rightarrow$ closest pair

Hull Feature Compatibility

A vertex and edge are *compatible* if some tangent at the vertex has same slope as the edge



Each edge has ≤ 2 compatible vertices
but vertex may have many compatible edges

Previous average-case algorithm:
maintain graph of all compatible pairs
but can have $\Omega(n)$ changes in worst case

Hull Feature Distance

Define distance between compatible features
to be Euclidean distance from point to line

But, define distance between incompatible features
to be infinite

> **Lemma:** The width of a point set equals the minimum distance between any two features

So width becomes a closest pair problem
Solve by reducing closest pairs to nearest neighbors

Nearest Compatible Feature

Find nearest vertex to edge

Binary search in hull for compatible vertex

(If more than one, both are equally close)

Find nearest edge to vertex

Binary search for the compatible edges
But what if there are many of them?

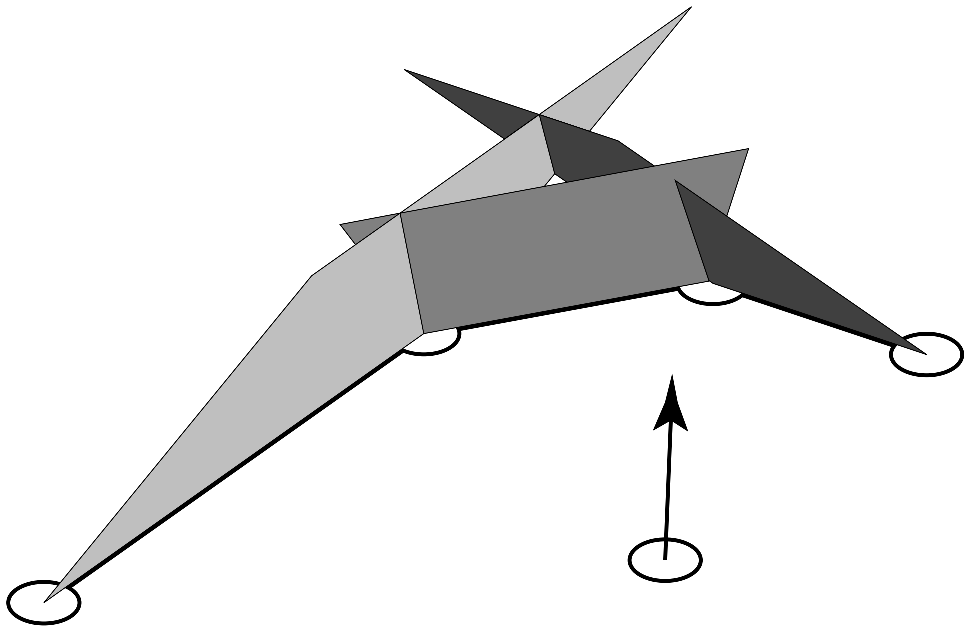
Use BB[α] tree to represent compatible edges
as union of $O(\log n)$ canonical sets

Search for nearest in each canonical set

Nearest Compatible Feature (continued)

Given canonical set of edges, all compatible with query vertex, which is the closest?

Cut each hull edge by equal slope planes in $\mathbb{R}^3$, then distance to edge = vertical distance to plane



Query becomes vertical ray shooting!

Vertical Ray Shooting

Given set of planes above query point

Which plane has smallest vertical distance to query?

Static:
Find intersection of halfspaces
Build point location data structure
$O(\log n)$ per query

Dynamic: $O(n^{\epsilon})$ per update or query
[Agarwal and Matoušek 1995]

We need the dynamic version

Closest Pair from Nearest Neighbors

Theorem: If we can perform dynamic nearest neighbor queries in time $T(n)$ per update or query, then we can maintain closest pairs in $T(n)\log^2 n$ per update.
[Eppstein 1995]

Even easier method (not in proceedings):

Each vertex remembers nearest compatible edge

When an update changes an edge, use neighbor query to find affected vertex

For each new or affected vertex, look up new nearest edge

Conclusions

Reduce width $\rightarrow$ vertical ray shooting
$O(\log n)$ ray shooting operations per hull change
so $O(n^{\epsilon})$ time per hull change

For incremental or decremental width,
$O(n^{\epsilon})$ time per point insertion or deletion

What about fully dynamic width?
Worst case instead of amortized time?

Generalizations to other problems
e.g. to min max distance problems?