Convex polyhedra
The **convex hull** of a finite set of points $p_i$ in 3d is:

- all convex combinations of points, where a convex combination is a weighted average $(\sum w_i p_i)/(\sum w_i)$ for non-negative weights $w_i$
- the union of tetrahedra defined by subsets of four points $p_i$
- the intersection of all half-spaces that contain the points $p_i$

[Pbierre 2015]
The faces of a 3d polyhedron are its vertices, edges, and facets.

Vertices are 0-dimensional, edges are 1d, facets are 2d.

(Sometimes the facets are called faces, but we need a word for all of these things.)

It is convenient to think of the empty set (dimension = $-1$) and the whole polyhedron (dimension = 3) as also being faces.

Face = any intersection of the polyhedron with a closed halfspace whose boundary is disjoint from the interior of the polyhedron.

(This generalizes to $d$-dimensional convex hulls: “polytopes”.)
The face lattice

Faces of all dimensions and their inclusion relations

Convex hull problem: Convert set of points to this structure
Euler’s formula

For 3d convex polyhedra: \( V - E + F = 2 \)

Or, in terms of face lattice with empty set and whole polyhedron

\[
\sum_{\dim=-1}^{d} (-1)^{\dim} \times (# \text{ faces of dim}) = 0
\]

This formula works for convex hulls in any dimension!

For instance, in 2d, it says: # vertices = # edges
One of many proofs

- Rotate polyhedron so each face has exactly one vertex with minimum $z$-coordinate, and exactly one with maximum $z$-coordinate
- Place positive charge at each vertex, negative on each edge, positive in each facet, so total charge is $V - E + F$
- Rotate charges around $z$-axis into nearby facets
- Each facet gets $-1$ total from a path of edges and vertices along one side, cancelling its own $+1$ charge
- The only charges left are $+1$ at the north and south poles
3d hulls have linear complexity

\[ V - E + F = 2 \]

\[ 2E \geq 3F \]

(There are 2 faces per edge and at least three edges per face, so the number of face-edge incidences is exactly \(2E\) and at least \(3F\).)

...some algebra to eliminate \(E\) or \(F\) from Euler ...

\[ E \leq 3V - 6 = O(V) \]

\[ F \leq 2V - 4 = O(V) \]
Complexity blows up in higher dimensions

Worst-case convex hulls in \( d \) dimensions are for \( n \) points with coordinates \((x, x^2, x^3, \ldots, x^d)\) (for \( n \) different values of \( x \))

\[
\# \text{ faces is } \Theta \left( n^{\lfloor d/2 \rfloor} \right)
\]

An unsolved problem: Suppose you have a 4d convex hull of \( n \) points, and it also has \( O(n) \) three-dimensional faces. How many 1d edges and 2d polygons can it have?
Convex hull algorithms in general
Graham scan doesn’t work

Graham scan: add points in sorted order by one coordinate (say z)

Bad example: Start with a cone of $n/2$ vertices and then keep making the peak higher.

Each added peak makes $n/2$ new faces $\Rightarrow$ total time $\Theta(n^2)$
### Some ideas that do work

<table>
<thead>
<tr>
<th>Like mergesort</th>
<th>Like quicksort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Split arbitrarily into two subsets of ( n/2 ) points</td>
<td>Split at median of sorted order by coordinates</td>
</tr>
<tr>
<td>Recurse</td>
<td>Recurse</td>
</tr>
<tr>
<td>Merge the two hulls</td>
<td>“Gift-wrap” the two disjoint hulls</td>
</tr>
</tbody>
</table>

... but the details are complicated ...
The algorithm from the book
Main idea and data structures

Main idea: randomized incremental (add points in random order)

Maintain DCEL of hull vertices, edges, and facets

Also maintain “conflict graph”:

- Vertices: points that have not been added, and facets of current hull
- Edges: pairs of a point and a facet that it can see
- Represent as list of visible facets for each point, and list of conflict points for each facet
How to initialize the data structures

**Initial hull**
Choose two arbitrary points
Find a third point not in line with them
Find a fourth point not on a plane with them
Result is a tetrahedron

**Conflict graph**
Discard the remaining points that are inside the tetrahedron
For each remaining point, test which tetrahedron facets it can see (3d orientation primitive) and add those edges to the conflict graph
Main algorithm

For each remaining point $p$, in a random order:

- Use the conflict graph to find and remove all hull vertices that it can see, cutting a hole from the surface of the old hull
- For each edge $e$ of the hole, make a new triangle $T_e$ connecting $e$ to $p$
- If $T_e$ is in the same plane as the other facet on edge $e$, merge them
- Otherwise, for each point $q$ in the conflict lists of the two old facets on edge $e$, check whether $q$ should be added to the conflict list for $T_e$
Partial analysis

After adding point $i$, what is expected number of edges we just added?

- Current set of $i$ points has $\leq 3i - 6$ edges
- Edge $e$ was just added if we just added one of its 2 endpoints
- Because of the random permutation, each of the $i$ points is equally likely to be the one we just added
- So probability we just added edge $e$ is $2/i$

Expected number of new edges $\leq (3i - 6) \cdot \frac{2}{i} = 6 - \frac{12}{i} < 6$

(linearity of expectation)
Partial analysis

Expected \# new edges in each step is < 6
\Rightarrow expected \# edges for whole algorithm is < 6n
(linearity of expectation again)

So total expected change to DCEL, over whole algorithm, is $O(n)$

Harder part: expected time for updating conflict lists is $O(n \log n)$
Pbierre. 3D convex hull of a 120 point cloud. Licensed under the Creative Commons Attribution-Share Alike 4.0 International license, October 19 2015. URL https://commons.wikimedia.org/wiki/File:3D_Convex_Hull.tiff.