# Convexe Hull in 3-Space

Jonas Sorgenfrei – Minf101767

**Algorithm** Convex Hull(*P*)

*Input*: A set *P* of *n* points (x,y,z) in three-space.

*Output*: The convex hull *CH(P)* of *P*.

1. Find 4 points *p1, p2, p3, p4* in *P* that form a tetrahedron.

2. C 🡨 *CH*({*p1, p2, p3, p4*})

3. Compute a random permutation *p5, p6, …, pn* of the remaining points

4. Initialize the conflict graph *G* with all visibile pairs (*pt, f*), where *f* is a facet of C and *t* > 4

5. **for** *r* = 5; *r* < *n*; *r*++

6. **do** // Insert *pr* into *C*:

7. **if** *Fconflict(pr)* is not empty // *pr* lies outside of C

8. **then** Delete all facets in *Fconflict(pr)* from *C*

9. Walk along the boundary of the visible region of *pr* (which consists exactly of the facets in *Fconflict(pr)*) and create a list *L* of horizon edges in order.

10. **for** all e ∈ L

11. **do** Connect *e* to *pr* by creating a triangluar facet *f*

12. **if** *f* is coplanar with its neighbor *f‘* along *e*

13. **then** merge *f* and *f‘* into one facet, whose conflict list is the same as that of *f‘*

14. **else** // Determine conflicts for *f:*

15. Create a node for *f* in *G*.

16. Let *f1* and *f2* be the facets incident to *e* in the old convex hull.

17. *P(e)* 🡨 *Pconflict(f1) U Pconflict(f2)*

18. **for** all points *p ∈ P(e)*

19. **do** if *f* is visible from *p*,

add (p,f) to *G*

20. Delete the node corresponding to *pr* and the nodes corresponding to the facets in *Fconflict(pr)* from *G*, together with their incident arcs

21. **return** C

References:

Mark de Berg / Otfried Cheong / Marc van Kreveld / Mark Overmars: Computational Geometry, Algorithms and Applications Springer 2008 (3. Aufl.), ISBN 978-3-540-77973-5

https://github.com/jonassorgenfrei/convexHull3D