# Report for Computer GraphicII, HW1
# 3D convex hull algorithm and collision detection

XXX number

March 10, 2022

Acknowledgements: Deadline: 2022-03-10 23:59:59

You can choose C++ or Python, and no restrictions on programming framework. You can freely use frameworks such as openGL.

The **report** submits as a PDF file to gradscope, the programming part should package all the files include code, input files, executable file, readme.txt, and report. The **package** name is **your_student_name+student_id.zip**.

You will get Zero if the code not passing the plagiarism check.

# 1 Part 1 (20 points)

1. (5 points) Prove the intersection of two convex set is still a convex set.

   1) If the intersection is empty or consists of a point, the theorem is true by definition.

   2) Otherwise, take any two points, A, B in the intersection. The line AB connecting these points must also lie entirely within each of the sets and therefore must lie entirely within their intersection.

   REF: https://en.wikibooks.org/wiki/Convexity/The_intersection_of_convex_sets_is_convex

2. (15 points) If a plane is divided into polygons by line segments, please design a data structure to store the division information so that for the given line passing two points $p_1$ and $p_2$ on the plane, it is efficient to find all the polygons intersected with the line. Please provide the main idea and pseudocode of the algorithm and give the complexity analysis.

   No idea.

# 2 Part 2 (80 points)

## 2.1 3D convex hull algorithm(55 points)

The algorithm I choosed is incremental 3D convex hull algorithm. And for easier implementation of algorithms and visualizations, I choose a diction ,DCEL, as my data structure which could provide DCEL (dict-based), Vertex, hEdge, and Face classes. And the code of DCEL is adapted from list-based python 2.7 implementation at: https://github.com/Ylannl/pydcel.

1. Description

   1) As Fig.**??** shown, pick four points from the point set to form an initial tetrahedron.

   2) Select two points p1, p2, and then select a point p3, p3 and p1, p2 are not on the same line, thus forming a surface. Pick another point, not on the same face, to form a tetrahedron.

   3) Then move on to the remaining points:

   4) The point is inside the tetrahedron, skip it.

   5) The point is outside the tetrahedron, delete the faces that this point can "see", and expand the convex hull volume.

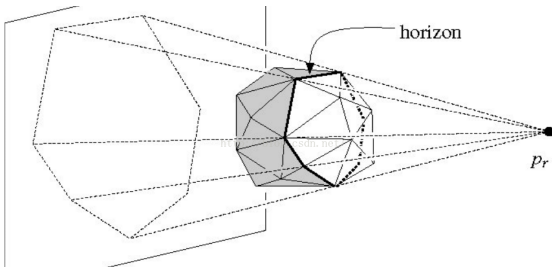


Figure 1: incremental 3D convex hull algorithm

2. Input data

   1) Using numpy to randomly generate the data.

   2) Use function np.random.randint (-100, 100,(100,3)). And the data could be float too.

3. Visualization

   As Fig.**??** shown.

4. Run time

   Time complexity: $O(n^2)$

   REF:Kenneth L. Clarkson, and Peter W. Shor. "Applications of random sampling in computational geometry, II." Discrete and Computational Geometry, vol.4, no.1, pp.387-421, 1989.

### 2.2 Collision detection(25 points)

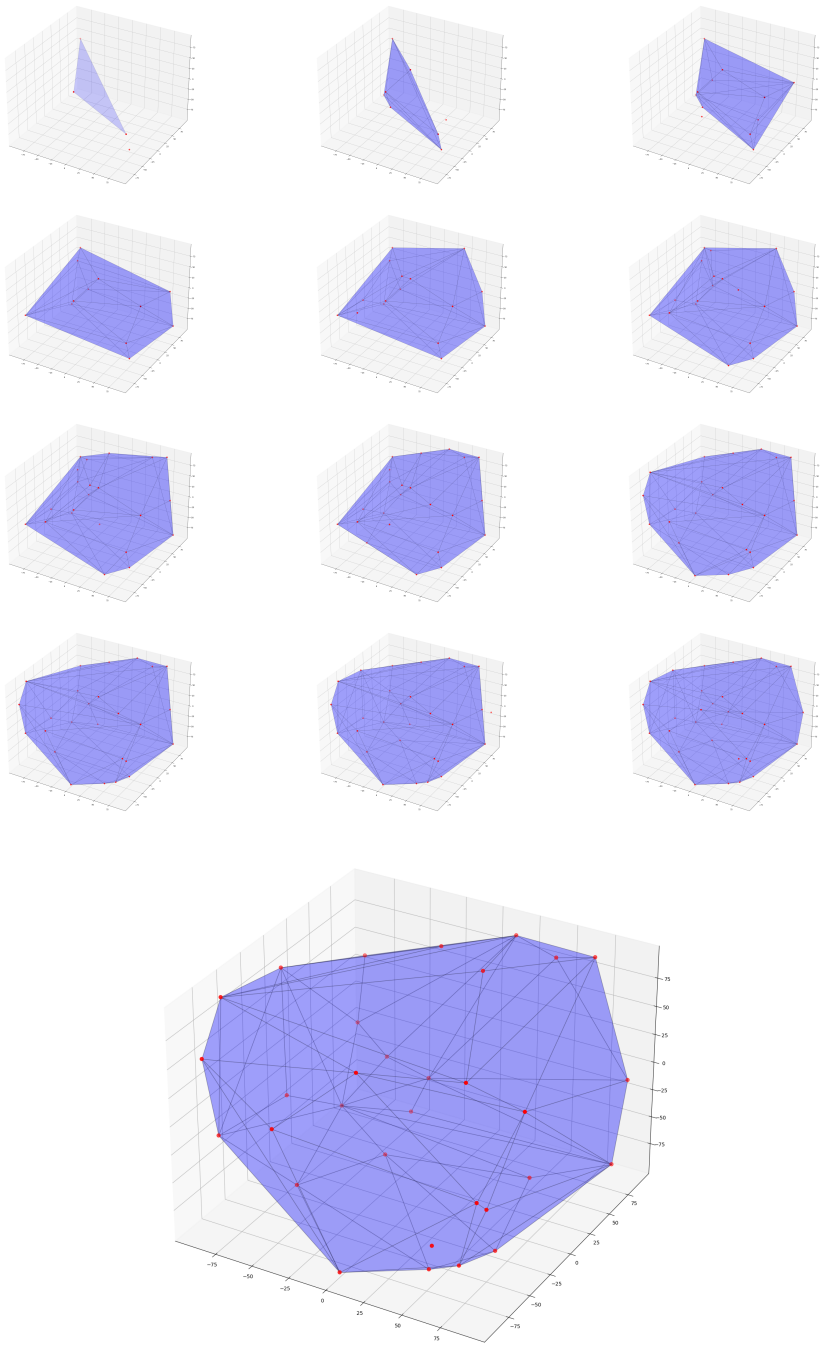

Figure 2: Visualization of generate the convex hull