Assignment One: Convex Hull Construction

David Gu

Yau Mathematics Science Center
Tsinghua University
Computer Science Department
Stony Brook University

gu@cs.stonybrook.edu

October 20, 2020

Convex Hull

The input to the convex hull algorithm is a set of 3D points

$$P = \{p_1, p_2, \ldots, p_n\}$$

The output is the convex hull of the point set P.

Input

The input points are randomly generated within the unit sphere.

Output

The convex hull is represented as a triangle mesh, using Dart data structure to store.

Convex Hull

Algorithm Pipeline

- Pick three points to form two triangles with opposite orientations, and glue them to form a topological ball, and assign the ball as the initial convex hull C;
- select a point p_k, which is as far as possible from the current C;
- For each face on the hull C, test the visibility with respect to p_k;
- Remove all the visible faces from C;
- For each edge $[p_i, p_j]$ on the contour (the curve separating the visition and invisible parts of C), connect the edge with the point p_k to form a triangle $[p_i, p_j, p_k]$, add the face to C;
- Repeat step 2 through 5, until all the points have been processed.

Convex Hull

Visibility Testing

Given a face $[p_i, p_j, p_k]$ and the new point p_l, the visibility testing is equivalent to compute the volume of the tetrahedron $[p_i, p_j, p_k, p_l]$, which is given by

$$\frac{1}{6}\begin{vmatrix} x_i & y_i & z_i & 1 \\ x_j & y_j & z_j & 1 \\ x_k & y_k & z_k & 1 \\ x_l & y_l & z_l & 1 \end{vmatrix}$$

and check whether is the volume is positive or not.

Example

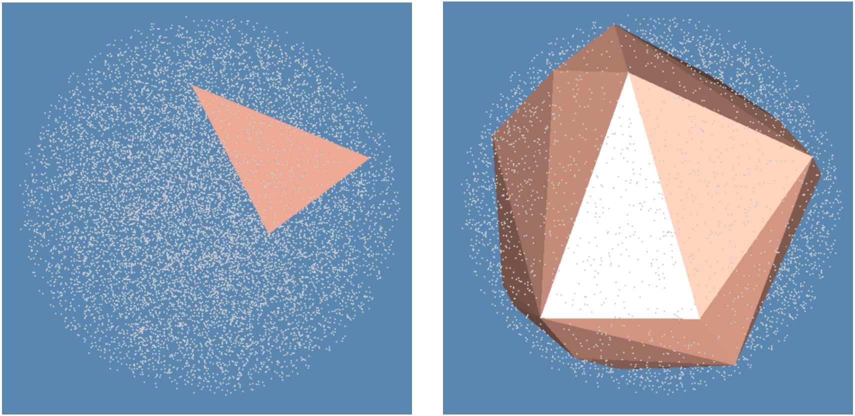

Figure: Convex hull computation process.

Example

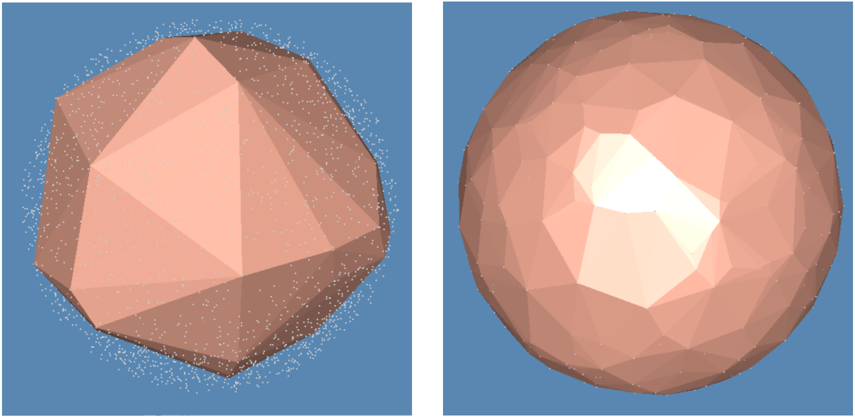

Figure: Convex hull computation process.

Instruction

Dependencies

1. ‘DartLib‘, a general purpose mesh library based on Dart data structure.
2. ‘freeglut‘, a free-software/open-source alternative to the OpenGL Utility Toolkit (GLUT) library.

Directory Structure

- 3rdparty/DartLib, header files for mesh;
- convex_hull/include, the header files for convex_hull;
- convex_hull/src, the source files for convex_hull;
- CMakeLists.txt, CMake configuration file;

Configuration

Before you start, read README.md carefully, then go three the following procedures, step by step.

1. Install [CMake](https://cmake.org/download/).
2. Download the source code of the C++ framework.
3. Configure and generate the project for Visual Studio.
4. Open the .sln using Visual Studio, and complie the solution.
5. Finish your code in your IDE.
6. Run the executable program.

Configure and generate the project

1. open a command window
2. cd Assignment_1_skeleton
3. mkdir build
4. cd build
5. cmake ..
6. open OTHomework.sln inside the build directory.

Finish your code in your IDE

- You need to modify the file: HandleTunnelLoop.cpp;
- search for comments "insert your code"
- Modify functions:
  1. $ConvexHull :: \_volume\_sign(CConvexHullMesh :: CFace*, constCPoint)$
  2. $ConvexHull :: \_inside(constCPoint)$
  3. $ConvexHull :: \_remove\_visible(constCPoint)$
  4. $ConvexHull :: \_close\_cap(constCPoint)$

Finish your code in your IDE

Modify assignment one, CutGraph, to implement the algorithms for null homologous cycle detection and Birkhoff curve shortening.