Triangle - Delaunay Triangulator

eryar@163.com

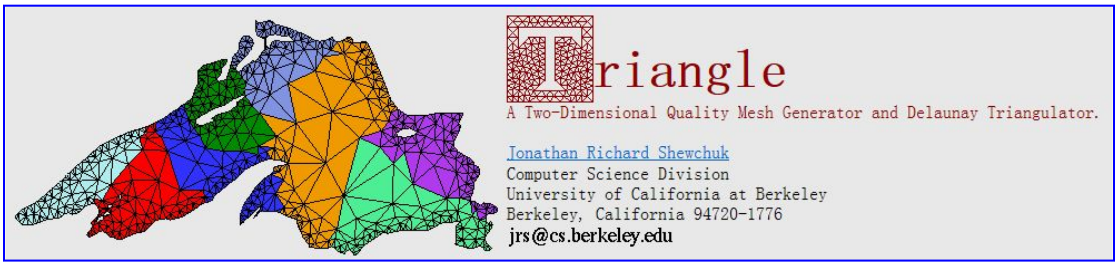
Abstract. Triangle is a 2D quality mesh generator and Delaunay triangulator. Triangle was created as part of the Quake project in the school of Computer Science at Carnegie Mellon University by Jonathan R. Shewchuk. Triangle is a small C program and its Delaunay refinement algorithm for quality mesh generation is a hybrid one. It includes divide-and-conquer and incremental insertion algorithms and sweepline Delaunay triangulation algorithm. This paper is focused on the usage of the Triangle and visualization the triangulation result in OpenSceneGraph.

Key words. Triangle, Delaunay Triangulator, Mesh Generator

1. Introduction

Triangle 可以生成精确的 Delaunay 三角剖分，限定 Delaunay 三角剖分（Constrained Delaunay Triangulation），Conforming Delaunay Triangulation，Voronoi 图（Voronoi Diagrams）和高质量的三角网格，即生成的网格中没有瘦长的三角形，所以适用于有限元分析（Finite Element Analysis）。

在 OpenCascade6.2.0 版本之前，OpenCascade 中网格的生成就是使用了这个开源库，由此可见 Delaunay 三角剖分算法和网格生成算法的重要性及广泛应用。



Triangle

A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator.

Jonathan Richard Shewchuk
Computer Science Division
University of California at Berkeley
Berkeley, California 94720-1776
jrs@cs.berkeley.edu

Figure 1.1 Triangle - A 2D Quality Mesh Generator and Delaunay Triangulator

下载 Triangle 的源程序及更多与 Triangle 相关信息的网址如下所示：

http://www.cs.cmu.edu/~quake/triangle.html

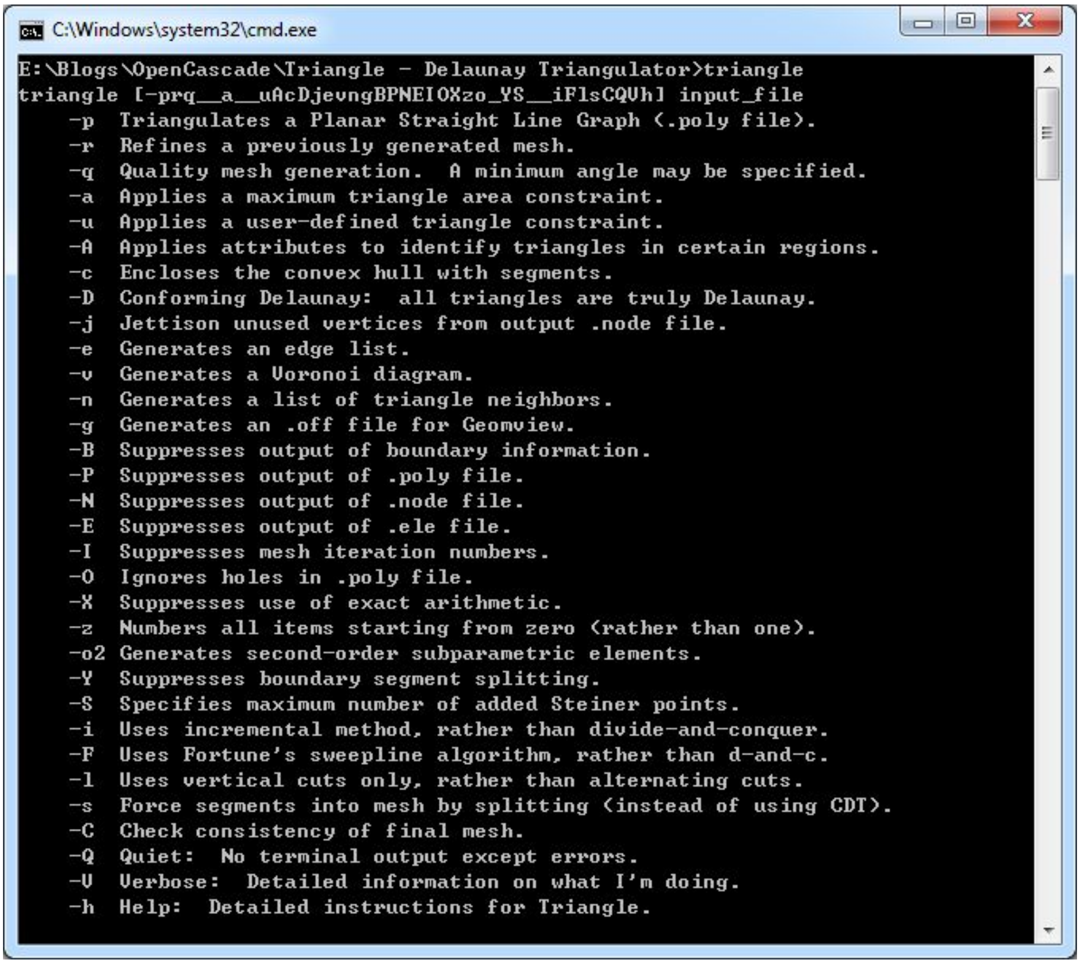
下载到源程序后，如果是 Windows 操作系统，还需要在 triangle.h 之前做些配置，如定义以下几个宏：

```
#define REAL double
#define ANSI_DECLARATORS
#include "triangle.h"
#undef REAL
```

在 triangle.c 中定义宏：#define NO_TIMER。有了上面的宏定义，可以编译出一个 triangle.exe 程序了。如果要将 triangle 用在自己的程序中，还需要定义#define TRILIBRARY。更多宏定义可以参考源程序。

2. Triangle Usage

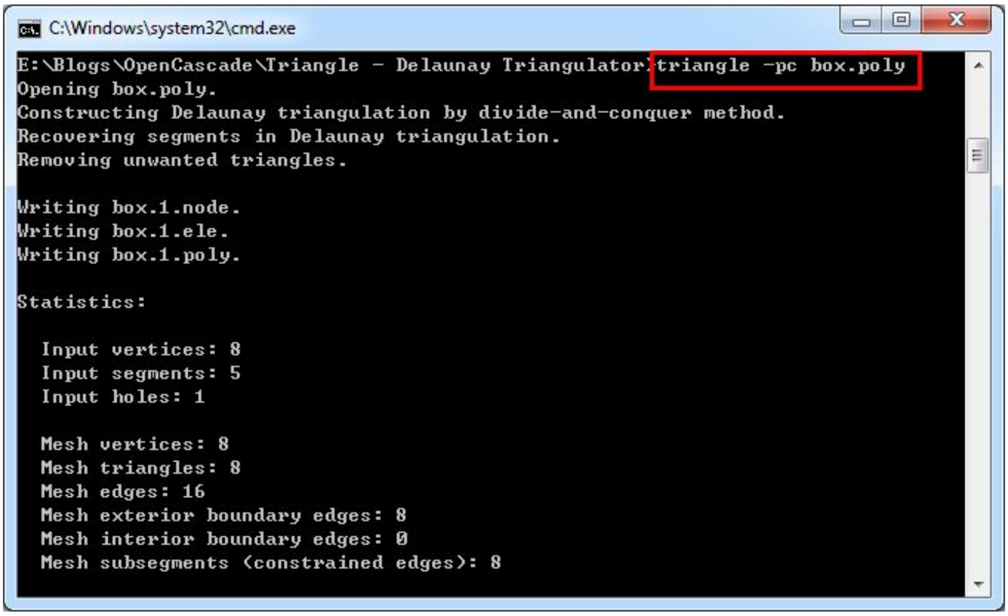
Triangle 有很多开关，可以选择三角剖分和生成网格的方式，如下图所示：



```
E:\Blogs\OpenCascade\Triangle - Delaunay Triangulator>triangle
triangle [-prq__a__uAcDjevngBPNEIOXzo_YS__iFlsCQVh] input_file
    -p  Triangulates a Planar Straight Line Graph (.poly file).
    -r  Refines a previously generated mesh.
    -q  Quality mesh generation.  A minimum angle may be specified.
    -a  Applies a maximum triangle area constraint.
    -u  Applies a user-defined triangle constraint.
    -A  Applies attributes to identify triangles in certain regions.
    -c  Encloses the convex hull with segments.
    -D  Conforming Delaunay:  all triangles are truly Delaunay.
    -j  Jettison unused vertices from output .node file.
    -e  Generates an edge list.
    -v  Generates a Voronoi diagram.
    -n  Generates a list of triangle neighbors.
    -g  Generates an .off file for Geomview.
    -B  Suppresses output of boundary information.
    -P  Suppresses output of .poly file.
    -N  Suppresses output of .node file.
    -E  Suppresses output of .ele file.
    -I  Suppresses mesh iteration numbers.
    -O  Ignores holes in .poly file.
    -X  Suppresses use of exact arithmetic.
    -z  Numbers all items starting from zero (rather than one).
    -o2 Generates second-order subparametric elements.
    -Y  Suppresses boundary segment splitting.
    -S  Specifies maximum number of added Steiner points.
    -i  Uses incremental method, rather than divide-and-conquer.
    -F  Uses Fortune's sweepline algorithm, rather than d-and-c.
    -l  Uses vertical cuts only, rather than alternating cuts.
    -s  Force segments into mesh by splitting (instead of using CDT).
    -C  Check consistency of final mesh.
    -Q  Quiet:  No terminal output except errors.
    -V  Verbose:  Detailed information on what I'm doing.
    -h  Help:  Detailed instructions for Triangle.
```

Figure 2.1 Options for the Triangle

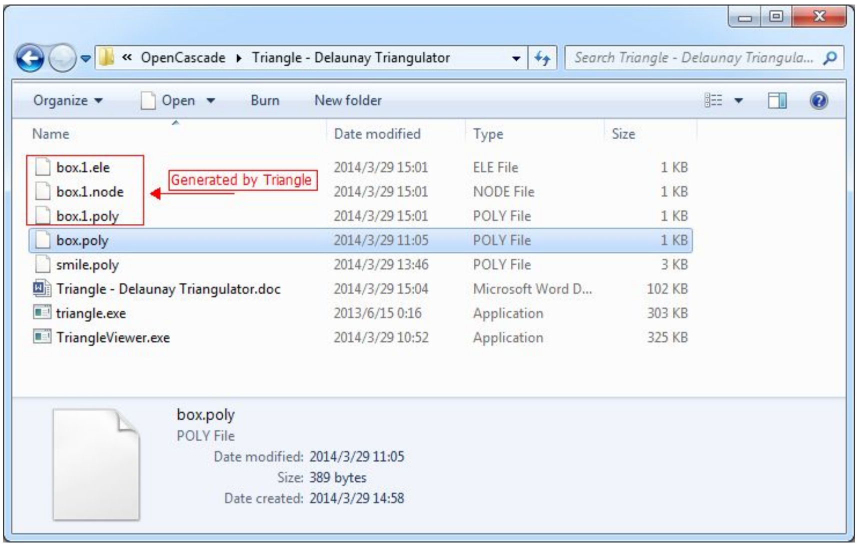
如对示例文件 box.poly 进行三角剖分，使用命令及生成结果统计信息如下所示：



```
E:\Blogs\OpenCascade\Triangle - Delaunay Triangulator>triangle -pc box.poly
Opening box.poly.
Constructing Delaunay triangulation by divide-and-conquer method.
Recovering segments in Delaunay triangulation.
Removing unwanted triangles.

Writing box.1.node.
Writing box.1.ele.
Writing box.1.poly.

Statistics:

  Input vertices: 8
  Input segments: 5
  Input holes: 1

  Mesh vertices: 8
  Mesh triangles: 8
  Mesh edges: 16
  Mesh exterior boundary edges: 8
  Mesh interior boundary edges: 0
  Mesh subsegments (constrained edges): 8
```

Figure 2.2 Triangle Usage

出现统计信息的同时也生成了一些文件，如顶点文件box.1.node和三角形文件box.1.ele，如下图所示：


« OpenCascade ▸ Triangle - Delaunay Triangulator
Search Triangle - Delaunay Triangula...
Organize ▾
Open ▾
Burn
New folder

Generated by Triangle

Name	Date modified	Type	Size
box.1.ele	2014/3/29 15:01	ELE File	1 KB
box.1.node	2014/3/29 15:01	NODE File	1 KB
box.1.poly	2014/3/29 15:01	POLY File	1 KB
box.poly	2014/3/29 11:05	POLY File	1 KB
smile.poly	2014/3/29 13:46	POLY File	3 KB
Triangle - Delaunay Triangulator.doc	2014/3/29 15:04	Microsoft Word D...	102 KB
triangle.exe	2013/6/15 0:16	Application	303 KB
TriangleViewer.exe	2014/3/29 10:52	Application	325 KB

box.poly
POLY File
Date modified: 2014/3/29 11:05
Size: 389 bytes
Date created: 2014/3/29 14:58

Figure 2.3 Nodes and Triangles data generated by Triangle

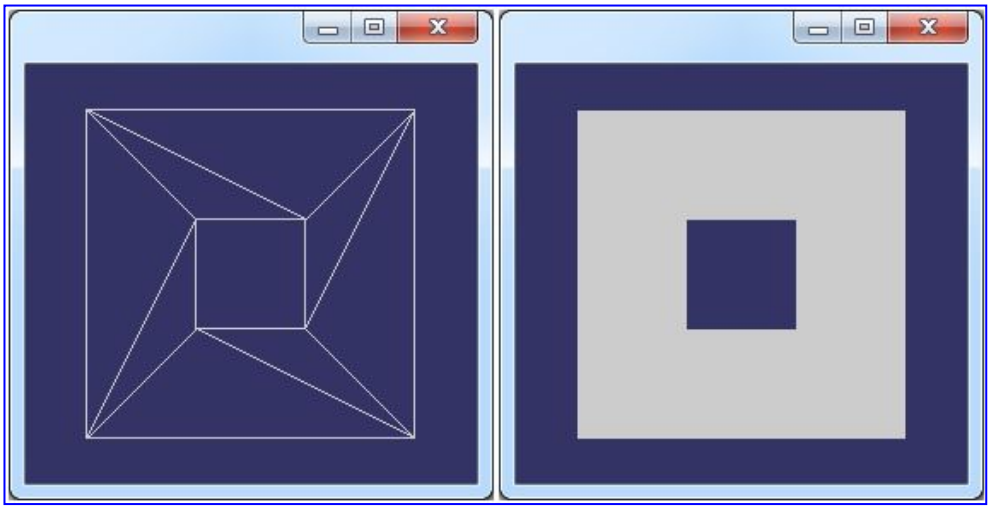


Figure 2.4 Triangulation Mesh Generated by Triangle[-pc]

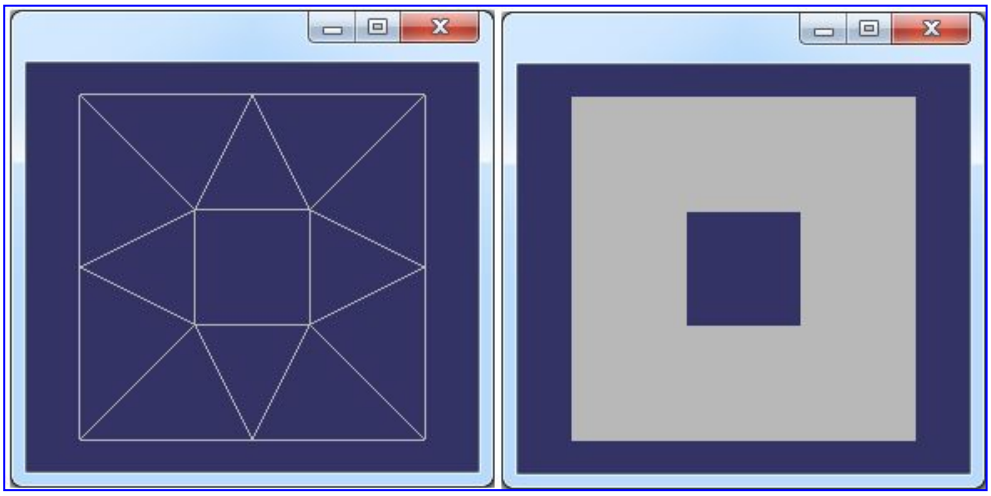
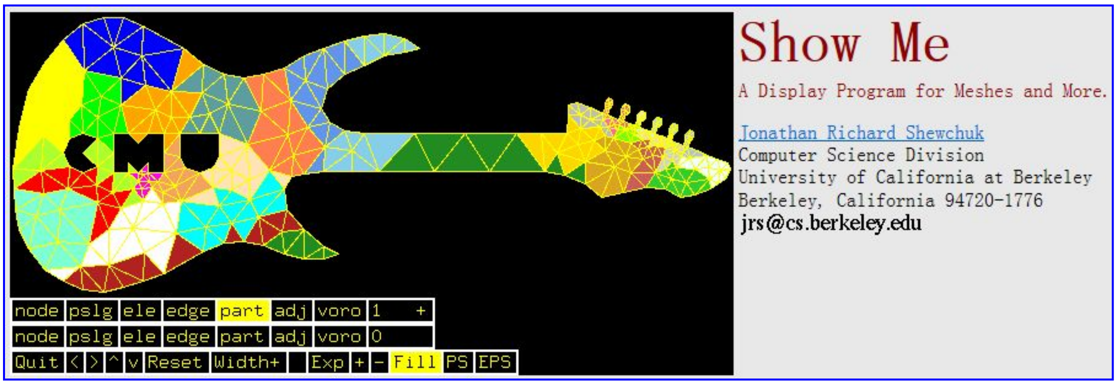


Figure 2.5 Triangulation Mesh Generated by Triangle[-pqc]

3. Displaying Meshes

在下载的程序中有用于显示网格的示例程序 showme.c，不过只能用于 Unix 操作系统，不能用于 Windows。



Show Me

A Display Program for Meshes and More.

Jonathan Richard Shewchuk
Computer Science Division
University of California at Berkeley
Berkeley, California 94720-1776
jrs@cs.berkeley.edu

CMU
node pslg ele edge part adj voro 1 +
node pslg ele edge part adj voro 0
Quit < > ^ v Reset Width+ Exp + - Fill PS EPS

Figure 3.1 Displaying the Meshes by ShowMe

为了在 Windows 操作系统中看到生成的网格，用 OpenSceneGraph 编写了一个小程序 TriangleViewer 显示网格。其中读取 node 和 element 文件中数据的主要程序片段如下所示：

```
std::string TriangleMesh::ReadLine(std::ifstream &theFile)
{
    std::string theBuffer;

    bool IsReadNextLine = false;

    do
    {
        getline(theFile, theBuffer);

        // skip comment here.
        if ('#' == theBuffer[0])
        {
            IsReadNextLine = true;
        }
        else
        {
            IsReadNextLine = false;
        }
    }
    while (IsReadNextLine);

    return theBuffer;
}

void TriangleMesh::BuildMesh(const std::string& aPolyFile)
{
    std::stringstream ss;

    std::string theNodeFileName(aPolyFile + ".node");
    std::string theElementFileName(aPolyFile + ".ele");
    std::ifstream theNodeFile(theNodeFileName.c_str());
    std::ifstream theElementFile(theElementFileName.c_str());

    Standard_Integer theIndex = 0;
    Standard_Integer theNodeCount = 0;
    Standard_Integer theTriangleCount = 0;

    Standard_Integer theIndex1 = 0;
    Standard_Integer theIndex2 = 0;
    Standard_Integer theIndex3 = 0;

    Standard_Real x = 0.0;
    Standard_Real y = 0.0;

    // Read mesh size.
    ss << ReadLine(theNodeFile);
    ss >> theNodeCount;

    ss.str("");
    ss.clear();

    ss << ReadLine(theElementFile);
    ss >> theTriangleCount;

    mMesh    =    new    Poly_Triangulation(theNodeCount,    theTriangleCount,
Standard_True);

    // Read nodes information.
    TColgp_Array1OfPnt2d& theNodes2d = mMesh->ChangeUVNodes();

    for (Standard_Integer n = 1; n <= theNodeCount; ++n)
    {
        ss.str("");
        ss.clear();

        ss << ReadLine(theNodeFile);
        ss >> theIndex >> x >> y;

        theNodes2d.SetValue(theIndex, gp_Pnt2d(x, y));
    }

    // Read triangles information.
    Poly_Array1OfTriangle& theTriangles = mMesh->ChangeTriangles();

    for (Standard_Integer t = 1; t <= theTriangleCount; ++t)
    {
        ss.str("");
        ss.clear();

        ss << ReadLine(theElementFile);
        ss >> theIndex >> theIndex1 >> theIndex2 >> theIndex3;

        theTriangles.SetValue(theIndex,   Poly_Triangle(theIndex1,   theIndex2,
theIndex3));
    }
}
```

如下图所示为显示一个用不同命令生成的 Smiley Face 的网格：

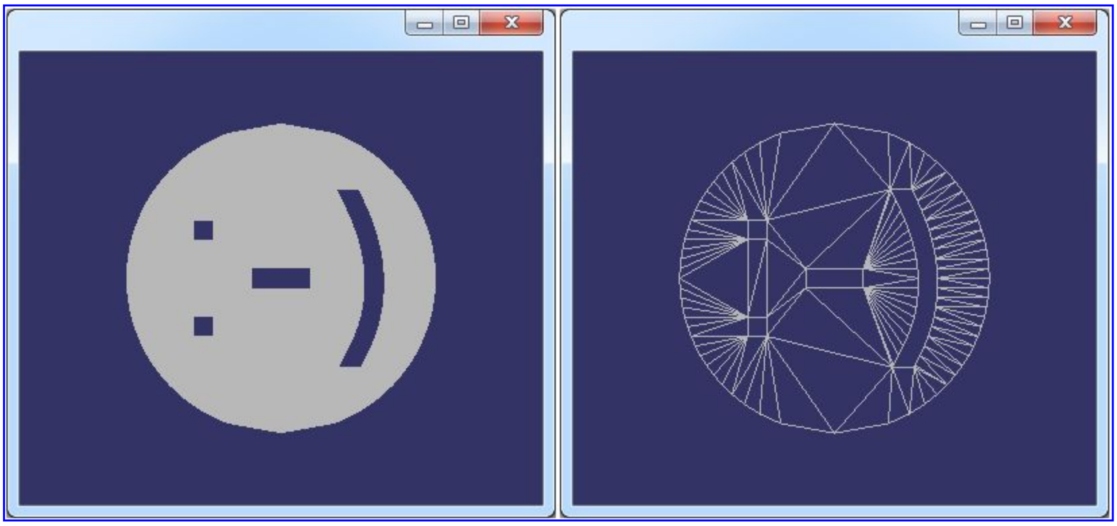


Figure 3.2 Generate Smiley Face Mesh by Triangle [-pc]

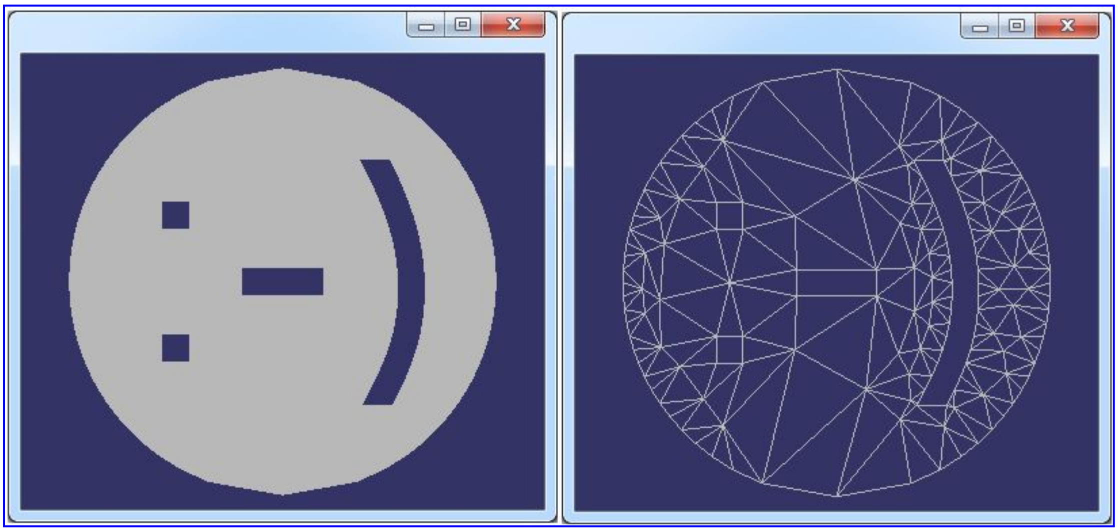


Figure 3.3 Generate Smiley Face Mesh by Triangle [-pqc]

从上面两幅图中的网格可知，下面图中的网格质量较高，为去掉了瘦长的三角形而增加了一些顶点。

4. Conclusions

在给 Triangle 程序输入数据时，顶点 Vertex 数据很好理解，只是一些二维点，但是如果加上开孔 Hole 后有些问题。后来才知道，需要在 Poly 文件中的 Segments 部分输入与孔相关线段形成的闭合区域，在孔 Hole 部分只需要输入位于孔中的任意一个点即可。

将 Triangle 生成的结果可视化，可以看到 Triangle 生成的网格，方便看到 Triangle 的不同选项生成的网格效果。

在 OpenCascade6.2.0 版本中，就以此二维 Delaunay 三角剖分工具为基础，实现了任意三维曲面的三角剖分，进而对其可视化。所以学习 Triangle 的用法，结合 OpenCascade 的源程序便于理解任意曲面的可视化实现的方法。

对 Delaunay 三角剖分算法感兴趣的读者，可以参考相关书籍[3],[4],[5],[6]。

5. References

1. Jonathan R. Shewchuk. Triangle: http://www.cs.cmu.edu/~quake/triangle.html
2. Jonathan R. Shewchuk, Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangualtor, Springer-Verlag, Berlin, 1996
3. 汪嘉业 王文平 屠长河 杨承磊. 计算几何及应用. 科学出版社. 2011
4. 王成恩. 面向科学计算的网格划分与可视化技术. 科学出版社. 2011
5. 周培德. 计算几何－算法设计与分析. 清华大学出版社. 2008
6. Berg M D 著 邓俊辉译. 计算几何－算法与应用. 清华大学出版社. 2009