Netgen mesh library : nglib

eryar@163.com

摘要 Abstract：本文主是对 Netgen 的库 nglib 的用法进行介绍。主要参考资料是 Netgen 用户指南。最后给出一个具体程序实例。

关键字 Key Words：Netgen, nglib, Mesh

一、引言 Introduction

Netgen 网格生成库 nglib 是以 C++源程序形式提供，可以编译为 Unix/Linux 或 Windows 上的库文件。程序开发使用的接口文件是 nglib.h。

二、头文件 The Header File

接口文件中包含了一些类型定义和函数调用，所有的 Netgen 类型和函数都带有前缀 Ng。类型和函数首字母大写，所有常量都是大写。

三、类型和常量 Types and Constants

```
// ** Constants used within Netgen *********************
/// Maximum allowed number of nodes per volume element
#define NG_VOLUME_ELEMENT_MAXPOINTS 10

/// Maximum allowed number of nodes per surface element
#define NG_SURFACE_ELEMENT_MAXPOINTS 8

// *** Data-types for accessing Netgen functionality ***
/// Data type for NETGEN mesh
typedef void * Ng_Mesh;

/// Data type for NETGEN CSG geometry
typedef void * Ng_CSG_Geometry;

/// Data type for NETGEN 2D geometry
typedef void * Ng_Geometry_2D;

/// Data type for NETGEN STL geometry
typedef void * Ng_STL_Geometry;

// *** Special Enum types used within Netgen ***********
/// Currently implemented surface element types
enum Ng_Surface_Element_Type
   { NG_TRIG = 1, NG_QUAD = 2, NG_TRIG6 = 3, NG_QUAD6 = 4, NG_QUAD8 = 5 };

/// Currently implemented volume element types
enum Ng_Volume_Element_Type
   { NG_TET = 1, NG_PYRAMID = 2, NG_PRISM = 3, NG_TET10 = 4 };
```

```
/// Values returned by Netgen functions
enum Ng_Result
   {
     NG_ERROR               = -1,
     NG_OK                  = 0,
     NG_SURFACE_INPUT_ERROR = 1,
     NG_VOLUME_FAILURE      = 2,
     NG_STL_INPUT_ERROR     = 3,
     NG_SURFACE_FAILURE     = 4,
     NG_FILE_NOT_FOUND      = 5
   };

// *** Classes required for use within Netgen **********
/// Netgen Meshing Parameters class
class Ng_Meshing_Parameters
{
public:
   int uselocalh;                      //!< Switch to enable / disable usage of
local mesh size modifiers
   double maxh;                        //!< Maximum global mesh size allowed
   double minh;                        //!< Minimum global mesh size allowed

   double fineness;                    //!< Mesh density: 0...1 (0 => coarse; 1 =>
fine)
   double grading;                     //!< Mesh grading: 0...1 (0 => uniform mesh;
1 => aggressive local grading)
   double elementsperedge;             //!< Number of elements to generate per edge
of the geometry
   double elementspercurve;             //!< Elements to generate per curvature
radius
```

Ng_Mesh 表示 Netgen 网格的数据结构。Ng_STL_Geometry 表示 STL 几何。可以通过 Ng_Meshing_Parameters 来指定生成网格时的参数。Netgen 函数的返回值类型是 Ng_Result。

四、初始化 Initialization

分别调用如下两个函数对 netgen 初始化和析构：

```
/*! \brief Initialise the Netgen library and prepare for use
*/
DLL_HEADER void Ng_Init ();
/*! \brief Exit the Netgen meshing kernel in a clean manner
*/
DLL_HEADER void Ng_Exit ();
```

五、网格的访问 Mesh access

Netgen 网格可以通过如下函数来处理。一个网格包含 nodes, surface elements 和 volume elements。都是从 1 开始计数的。

```
// Generates new mesh structure
```

```
Ng_Mesh * Ng_NewMesh ();
void Ng_DeleteMesh (Ng_Mesh * mesh);
// feeds points, surface elements and volume elements to the mesh
void Ng_AddPoint (Ng_Mesh * mesh, double * x);
void Ng_AddSurfaceElement (Ng_Mesh * mesh, Ng_Surface_Element_Type et, int * pi);
void Ng_AddVolumeElement (Ng_Mesh * mesh, Ng_Volume_Element_Type et, int * pi);
// ask for number of points, surface and volume elements
int Ng_GetNP (Ng_Mesh * mesh);
int Ng_GetNSE (Ng_Mesh * mesh);
int Ng_GetNE (Ng_Mesh * mesh);
// return point coordinates
void Ng_GetPoint (Ng_Mesh * mesh, int num, double * x);
// return surface and volume element in pi
Ng_Surface_Element_Type
Ng_GetSurfaceElement (Ng_Mesh * mesh, int num, int * pi);
Ng_Volume_Element_Type
Ng_GetVolumeElement (Ng_Mesh * mesh, int num, int * pi);
```

更多信息请参考其头文件 nglib.h。

在生成网格时，用户可以指定网格大小及一些约束。函数 Ng_GenerateVolumeMesh 从曲面生成网格。

```
/*! \brief Apply a global restriction on mesh element size
*/
DLL_HEADER void Ng_RestrictMeshSizeGlobal (Ng_Mesh * mesh, double h);

/*! \brief Locally restrict the mesh element size at the given point
*/
DLL_HEADER void Ng_RestrictMeshSizePoint (Ng_Mesh * mesh, double * p, double h);

/*! \brief Locally restrict the mesh element size within a specified box
*/
DLL_HEADER void Ng_RestrictMeshSizeBox (Ng_Mesh * mesh, double * pmin, double * pmax,
double h);
/*! \brief Create a 3D Volume Mesh given a Surface Mesh
*/
DLL_HEADER Ng_Result Ng_GenerateVolumeMesh (Ng_Mesh * mesh, Ng_Meshing_Parameters
* mp);
```

六、STL 几何 STL Geometry

STL 几何数据可以从 STL 文件（ASCII 或二进制）读取，也可以由一个个的三角形组装而成。

```
// loads geometry from STL file
DLL_HEADER Ng_STL_Geometry * Ng_STL_LoadGeometry (const char * filename, int binary
= 0);


// generate new STL Geometry
```

```
DLL_HEADER Ng_STL_Geometry * Ng_STL_NewGeometry ();


// fills STL Geometry
// positive orientation
// normal vector may be null-pointer
DLL_HEADER void Ng_STL_AddTriangle (Ng_STL_Geometry * geom,
                         double * p1, double * p2, double * p3,
                         double * nv = NULL);

// add (optional) edges :
DLL_HEADER void Ng_STL_AddEdge (Ng_STL_Geometry * geom,
                     double * p1, double * p2);

// after adding triangles (and edges) initialize
DLL_HEADER Ng_Result Ng_STL_InitSTLGeometry (Ng_STL_Geometry * geom);

// automatically generates edges:
DLL_HEADER Ng_Result Ng_STL_MakeEdges (Ng_STL_Geometry * geom,
                            Ng_Mesh* mesh,
                            Ng_Meshing_Parameters * mp);


// generates mesh, empty mesh must be already created.
DLL_HEADER Ng_Result Ng_STL_GenerateSurfaceMesh (Ng_STL_Geometry * geom,
                                                 Ng_Mesh * mesh,
                                                 Ng_Meshing_Parameters * mp);
```

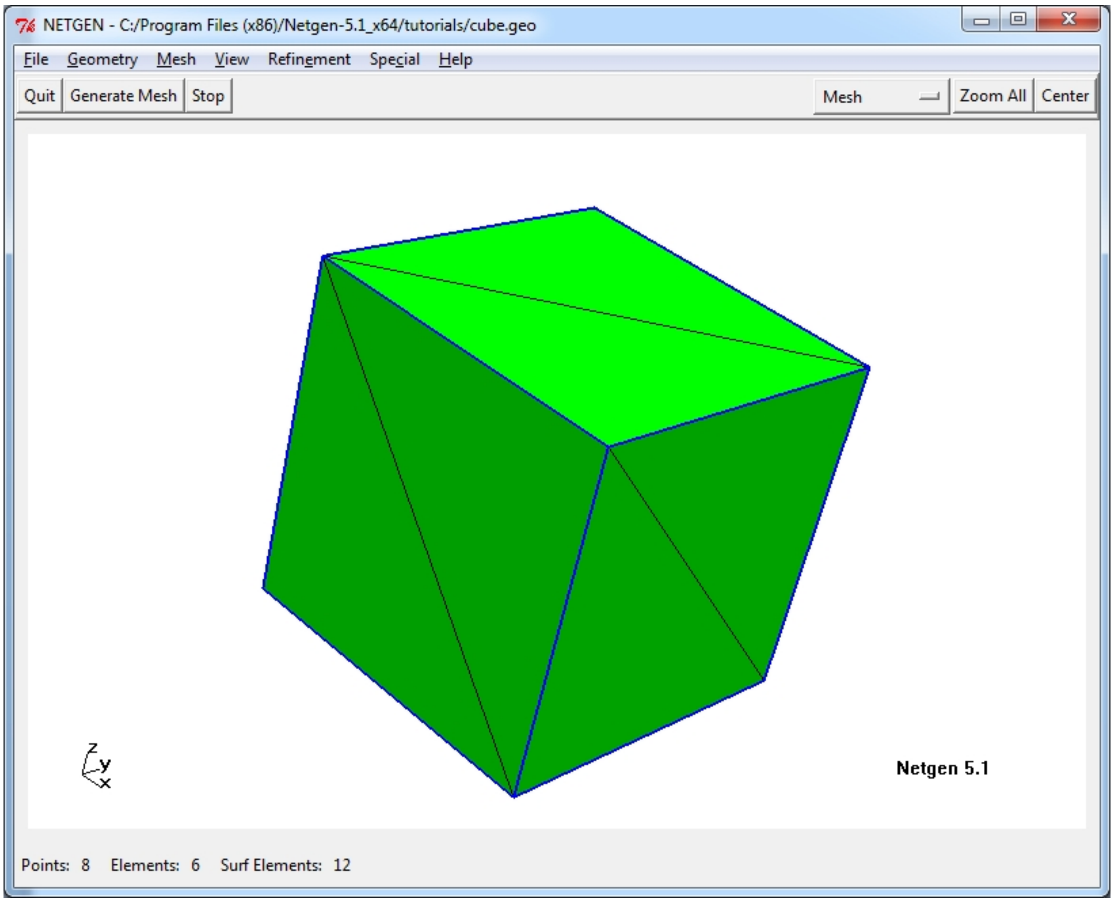
七、示例程序 Example Code

在 Netgen 的安装目录下的 nglib 文件夹中有两个示例程序 ng_vol.cpp 和 ng_stl.cpp。其中 ng_vol.cpp 程序开始时从 cube.surf 文件读入点坐标和曲面的三角形，生成的 volumne mesh 输出到文件：cuve.vol。经过测试，程序可以运行，修改过的代码如下所示：

```
#include <iostream>
#include <fstream>

namespace nglib {
#include <nglib.h>
}

#pragma comment(lib, "nglib.lib")

int main(int argc, char* argv[])
{
	std::cout << "Netgen Testing..." << std::endl;

	int i = 0;
	int np = 0;
	int nse = 0;
	int ne = 0;
	int trig[3] = {0};
	int tet[4] = {0};
```

```cpp
	double point[3] = {0.0};

	std::string strMeshFile = (argc > 1)? argv[1]: "cube.surf";
	std::ifstream meshFile(strMeshFile.c_str());

	// initialize the Netgen library.
	nglib::Ng_Init();

	// Generate new mesh structure.
	nglib::Ng_Mesh* mesh = nglib::Ng_NewMesh();

	// Read surface mesh from file.
	// feed points to the mesh.
	meshFile >> np;
	std::cout << "Reading " << np << " points..." << std::endl;
	for (int i = 0; i < np; ++i)
	{
		meshFile >> point[0] >> point[1] >> point[2];
		nglib::Ng_AddPoint(mesh, point);
	}
	std::cout << "done." << std::endl;

	// feed surface elements to the mesh.
	meshFile >> nse;
	std::cout << "Reading " << nse << " faces..." << std::endl;
	for (int i = 0; i < nse; ++i)
	{
		meshFile >> trig[0] >> trig[1] >> trig[2];
		nglib::Ng_AddSurfaceElement(mesh, nglib::NG_TRIG, trig);
	}
	std::cout << "done." << std::endl;

	// generate volume mesh.
	nglib::Ng_Meshing_Parameters mp;
	mp.maxh = 1e6;
	mp.fineness = 1;
	mp.second_order = 0;

	std::cout << "start meshing..." << std::endl;
	nglib::Ng_GenerateVolumeMesh(mesh, &mp);
	std::cout << "meshing done." << std::endl;

	// volume mesh output.
	np = nglib::Ng_GetNP(mesh);
	std::cout << "Points: " << np << std::endl;
	for (int i = 1; i <= np; ++i)
	{
		nglib::Ng_GetPoint(mesh, i, point);
		std::cout << i << ": " << point[0] << ", " << point[1] << ", " << point[2] << std::endl;
	}

	ne = nglib::Ng_GetNE(mesh);
	std::cout << "Elements: " << ne << std::endl;
	for (int i = 1; i <= ne; ++i)
	{
		nglib::Ng_GetVolumeElement(mesh, i, tet);
		std::cout << i << ": " << tet[0] << ", " << tet[1] << ", " << tet[2] << ", " << tet[3] <<
```

```
std::endl;
	}

	// Save mesh.
	nglib::Ng_SaveMesh(mesh, "test.vol");

	// deconstruct Netgen library.
	nglib::Ng_Exit();

	return 0;
}
```



Points: 8 Elements: 6 Surf Elements: 12