OpenGL图形编程

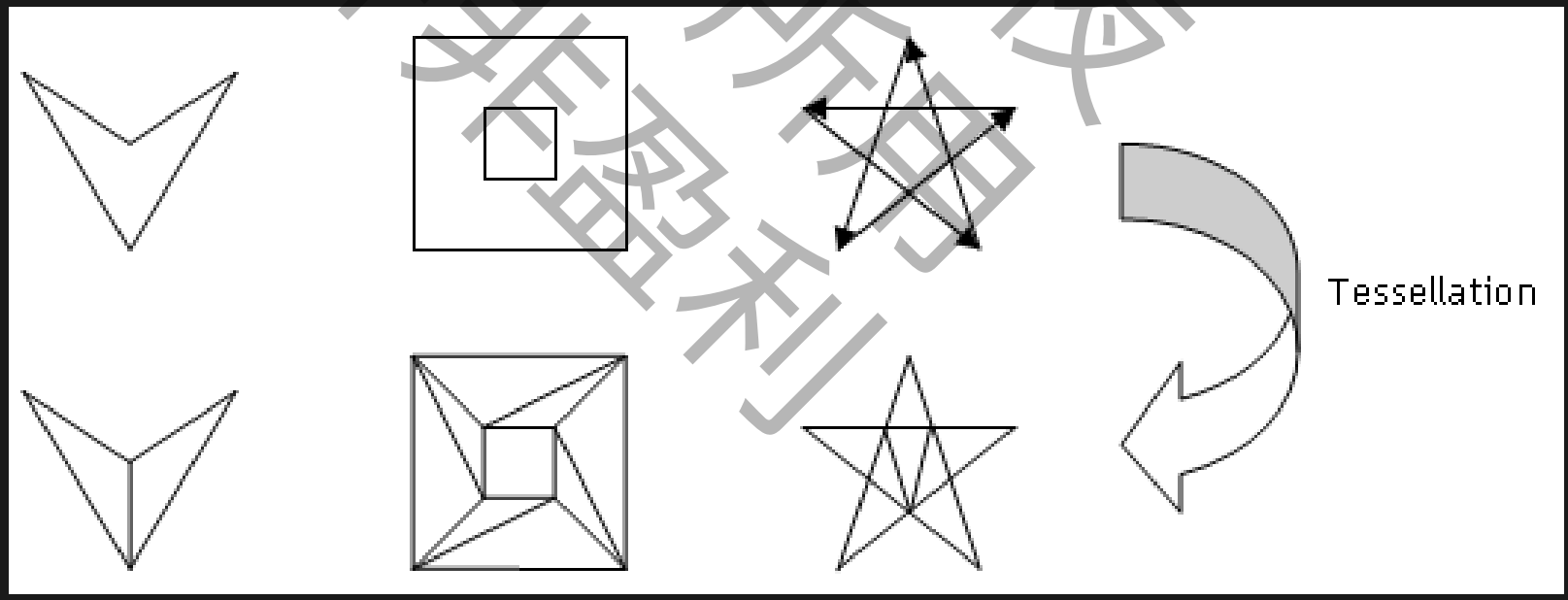
授课教师：少书师夜

4.栅格化曲线曲面与实体造型

- 4.1栅格化
- 4.2曲线曲面
- 4.3实体

4.1栅格化

- OpenGL只能直接显示简单的凸多边形。简单凸多边形就是多边形的边只在顶点处相交，没有重复的顶点，并且任何顶点都只有两条边相遇。
- 如果需要显示凹多边形、中间有洞的多边形或者具有相交边的多边形，就必须分解为简单的凸多边形，即栅格化。



4.1栅格化

- 复杂多边形栅格化步骤：
  - 创建一个栅格化对象
  - 注册在栅格化期间执行操作的回调函数
  - 指定栅格化属性
  - 指定一个或多个封闭多边形组成的轮廓，以创建并渲染分割后的多边形
  - 删除栅格化对象

4.1栅格化

- 创建栅格化对象

```
GLUtesselator* gluNewTess(void);
```

函数创建一个栅格化对象，并返回一个指向该对象的指针，如果创建失败则返回NULL指针。

4.1栅格化

- 栅格化回调函数

```
void gluTessCallback(GLUtesselator*
    tessobj,GLenum type,void (* fn)() );
```

函数将回调函数fn与栅格化对象tessobj关联起来，类型由参数type指定。

4.1栅格化

- 栅格化属性

```
void gluTessProperty(GLUtesselator
  *tessobj,GLenum property,GLdouble value );
```

函数设置栅格化对象tessobj的property属性，其值设为value。

最重要最复杂的是环绕规则，决定了多边形的“内部”和“外部”。

4.1栅格化

- 定义多边形

```
void glTessBeginPolygon(GLUtesselator *tessobj,void *user_data );
void glTessEndPolygon(GLUtesselator *tessobj );
```

此对函数将多边形同栅格对象tessobj关联起来，参数user_data指向用户定义的数据。可以定义一条或多条轮廓线。

```
void glTessBeginContour(GLUtesselator *tessobj);
void glTessEndContour(GLUtesselator *tessobj );
```

此对函数指定一条封闭的轮廓线，默认将轮廓线中最后一个顶点和第一顶点相连。

```
void gluTessVertex(GLUtesselator *tessobj,GLdouble
    coords[3],void *vertex_data);
```

函数指定轮廓线的一个顶点。

4.1栅格化

- 删去栅格化对象

```
void gluDeleteTess(GLUtesselator *tessobj );
```

删去栅格化对象tessobj,并释放其占用的内存。

4.1栅格化

- GLUtesselator *tess = gluNewTess(); // *创建网格器*
- // *注册回调函数*
- gluTessCallback(tess, GLU_TESS_BEGIN, beginCB);
- gluTessCallback(tess, GLU_TESS_END, endCB);
- gluTessCallback(tess, GLU_TESS_VERTEX, vertexCB);
- gluTessCallback(tess, GLU_TESS_COMBINE, combineCB);
- gluTessCallback(tess, GLU_TESS_ERROR, errorCB);
- gluTessBeginPolygon(tess, user_data); // *描述非凸多边形的顶点*
- // *第一条回路*
- gluTessBeginContour(tess);
- gluTessVertex(tess, coords[0], vertex_data);
- ...
- gluTessEndContour(tess);
- // *第二条回路*
- ...
- gluTessEndPolygon(tess);
- gluDeleteTess(tess); // *处理完成后删除网格器*

4.2.1Bezier曲线曲面

- Bezier曲线

Bezier曲线的求值函数：

```
void glMap1{fd}(GLenum target, TYPE t1, TYPE t2, GLint
    stride, GLint order, const TYPE *points);
```

参数target给出控制点数组表示的内容，一般取GL_MAP1_VERTEX_3,表示控制点数组存储控制点的三维点坐标。

参数t1,t2表示Bezier曲线参数t的最小和最大值，一般为0.0和1.0。

参数stride表数组points中一个坐标位置到另一个坐标位置的偏移量，对三维坐标数组，stride=3。

参数order指定Bezier曲线的阶数。

参数points为指向控制点数组的指针。

4.2.1Bezier曲线曲面

- Bezier曲线的生成步骤

1.指定控制点坐标；

2.设定求值函数；

3.激活求值函数

```
glEnable(GL_MAP1_VERTEX_3)
```

4.计算沿样条路径的位置并显示曲线

```
void glEvalCoord1{fd}(TYPE t);
```

4.2.1Bezier曲线曲面

- Bezier曲线的生成步骤

第4步还可以用以下函数来生成一组均匀分布的参数值，显示曲线

//指定曲线参数t从t1开始经过n步均匀地变为t2。

```
glMapGrid1{fd}(GLint n,TYPE t1,TYPE t2);
```

//指定从第n1个到第n2个参数（由glMapGrid1算出）绘制

```
glEvalMesh1(GLenum mode,GLint n1,GLint n2);
```

参数mode取值为GL_POINT或GL_LINE,表示以点或折线的形式显示曲线

4.2.1Bezier曲线曲面

- Bezier曲面

Bezier曲面的求值函数：

```
void glMap2{fd}(GLenum target, TYPE u1, TYPE u2, GLint ustride,
    GLint uorder, TYPE v1, TYPE v2, GLint vstride, GLint vorder, const
    TYPE *points);
```

参数u1,u2,v1,v2分别表示Bezier曲面参数u和v的最大最小值。

参数ustride和vstride分别表示数组points中相邻控制点的偏移量。

参数uorder和vorder指定Bezier曲面的阶数。

参数points为指向控制点坐标数组的指针。

4.2.1Bezier曲线曲面

- Bezier曲面的生成步骤

1.指定控制点坐标；

2.设定求值函数；

3.激活求值函数

```
glEnable(GL_MAP2_VERTEX_3)
```

4.计算沿样条路径的位置并显示曲面

```
glEvalCoord2{fd}(TYPE u, TYPE v);
```

4.2.1Bezier曲线曲面

- Bezier曲面的生成步骤

第4步还可以用以下函数来生成一组均匀分布的参数值，显示曲面

glMapGrid2{fd}(GLint nu,TYPE u1,TYPE u2, GLint nv,TYPE v1,TYPE v2);

glEvalMesh2(GLenum mode,GLint nu1,GLint nu2,GLint nv1,GLint nv2);

参数mode取值为GL_POINT、GL_LINE和GL_FILL,表示以点、折线或填充面的形式显示曲面

4.3实体

1.GLUT库中的多面体函数

表4.1 GLUT生成规则多面体的函数

函数	说明
glutSolidTetrahedron( ) glutWireTetrahedron( )	绘制中心位于世界坐标系原点的实心四面体和线框四面体，四面体的半径为$\sqrt{3}$ 。
glutSolidCube(size) glutWireCube(size)	绘制中心位于世界坐标系原点的实心立方体和线框立方体，立方体的半径为size，size是一个双精度浮点值。
glutSolidOctahedron ( ) glutWireOctahedron ( )	绘制中心位于世界坐标系原点的实心八面体和线框八面体，八面体的半径为1.0。
glutSolidDodecahedron( ) glutWireDodecahedron( )	绘制中心位于世界坐标系原点的实心12面体和线框12面体，12面体的半径为$\sqrt{3}$。
glutSolidIcosahedron( ) glutWireIcosahedron( )	绘制中心位于世界坐标系原点的实心20面体和线框20面体，20面体的半径为1.0。

4.3实体

- 2.GLUT库中的二、三次曲面
  - 绘制实体或线框球面

void glutSolidSphere/glutWireSphere (GLdouble radius, GLint slices, GLint stacks);

  - 绘制实体或线框圆锥面

void glutSolidCone/glutWireCone (GLdouble radius, GLdouble height, GLint slices, GLint stacks);

4.3实体

- 2.GLUT库中的二、三次曲面
  - 绘制实体或线框圆环

```
void glutSolidTorus/ glutWireTorus(GLdouble innerRadius,
    GLdouble outerRadius, GLint slices,GLint stacks);
```

  - 绘制实体或线框茶壶

```
void glutSolidTeapot/glutWireTeapot (GLdouble size);
```

4.3实体

- 3.GLU二次几何体

通过栅格化并用多边形逼近来模拟二维和三维几何体。

4.3实体

- 绘制二次几何体图元

球	gluSphere()
圆柱/圆锥/圆台	gluCylinder()
圆盘	gluDisk()
部分圆盘	gluPartialDisk()