山东大学 计算机科学与技术 学院

计算机网络 课程实验报告

学号：201700130011	姓名：刘建东	班级：17 级菁英班

实验题目：显示样条曲面

实验内容：

1. 使用 OpenGL 显示样条曲面
2. 显示曲面本身、控制多边形以及曲面的等参线

实验过程：

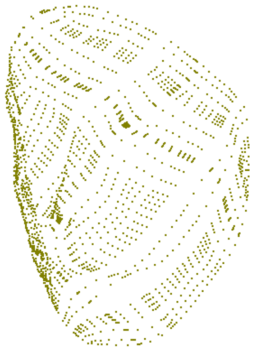
一、读入样条曲面文件

样条曲面文件中第一行是 n、m，表示控制顶点个数。第二行为 k1、k2，表示 B 样条曲面次数。接下来两行分别为 n、m 的节点向量。剩下的部分是 n*m 个控制点的三维坐标，第四维表示该点的权值。

用指针动态开辟空间，把上述这些信息读进来。

二、显示曲面控制点

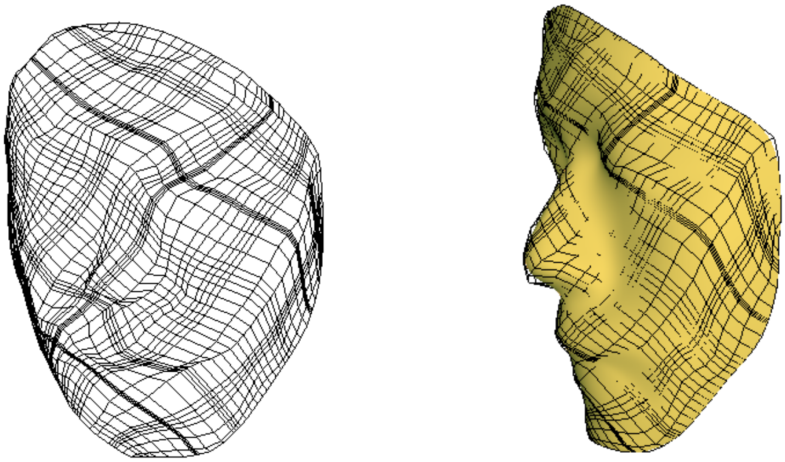
有了控制顶点坐标，显示控制点就非常简单，直接对于画出每一个点即可。



三、显示曲面控制多边形

接下来是显示曲面的控制多边形，其实控制多边形就是将 n、m 两个向量上的点连成一个网格，如果把 n、m 两个向量想成平面坐标系上的横纵坐标的话，控制多边形就是坐标系上的网格，因此先按 n 向量方向把点都连起来，再按 m 向量方向把点都连在一起即可。

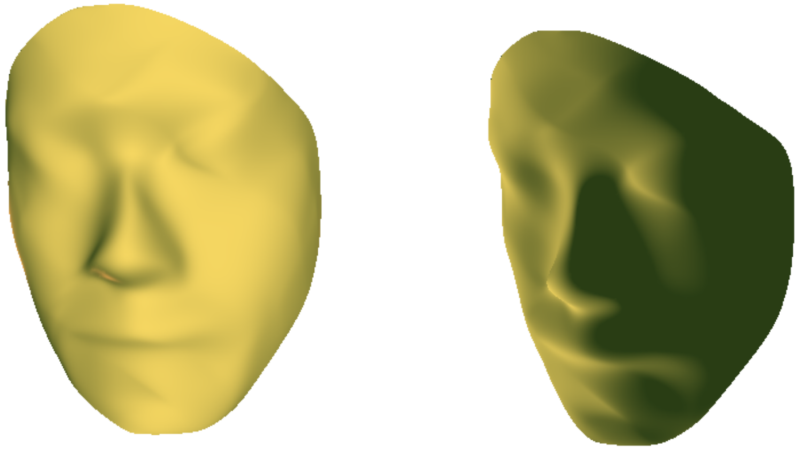
```
void show_control_polygon(){
    glDisable(GL_LIGHTING);
    rep(i,0,n[0]-1)
        rep(j,0,n[1]-2){
            glColor3f(0.0, 0.0, 0.0);
            glBegin(GL_LINES);
            glVertex3f(t_point[i][j][0], t_point[i][j][1], t_point[i][j][2]);
            glVertex3f(t_point[i][j+1][0], t_point[i][j+1][1], t_point[i][j+1][2]);
            glEnd();
        }

    rep(i,0,n[1]-1)
        rep(j,0,n[0]-2){
            glColor3f(0.0, 0.0, 0.0);
            glBegin(GL_LINES);
            glVertex3f(t_point[j][i][0], t_point[j][i][1], t_point[j][i][2]);
            glVertex3f(t_point[j+1][i][0], t_point[j+1][i][1], t_point[j+1][i][2]);
            glEnd();
        }
    glEnable(GL_LIGHTING);
}
```



可以看到控制多边形不是完全贴在曲面上的，即曲面是对控制多边形的一个逼近拟合。

四、显示曲面

显示曲面就直接调用 API 函数了，gluNurbsSurface 函数，根据函数定义，将参数一一填入即可完成曲面绘制，这里设置了光照，使得表面产生了阴影。



五、显示曲线等参线

显示等参线没有 API 可用，需要自己手动实现，基本原理是 u 值均匀取值，画出 u 值对应的每一条曲线，即可画出 v 向量方向上的等参线。使用下述公式手动实现。

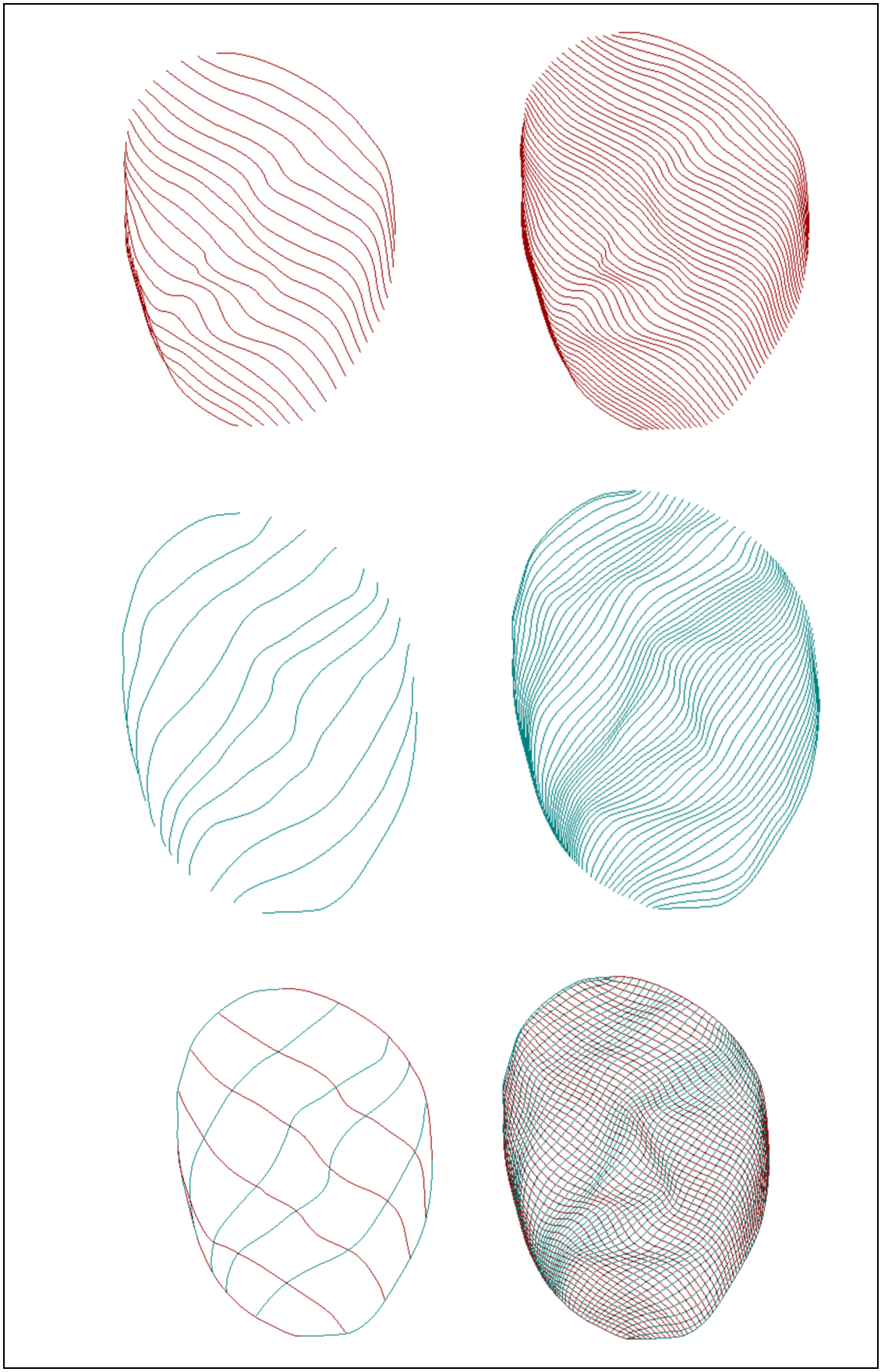
给定参数轴u和v的节点矢量

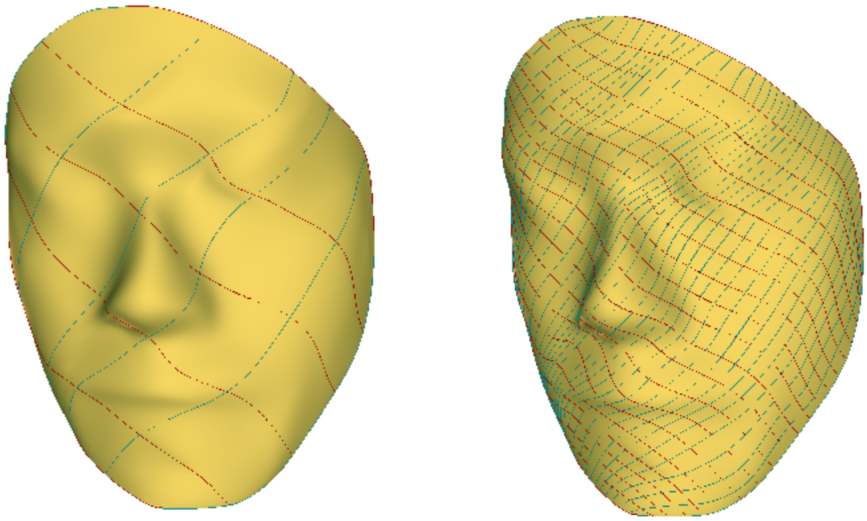
$$U = [u_0, u_1, \mathbf{L}\ , u_{m+p}]$$

$$V = [v_0, v_1, \mathbf{L}\ , v_{n+q}]$$

p×q阶B样条曲面定义如下：

$$P(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} P_{i,j} N_{i,p}(u) N_{j,q}(v)$$





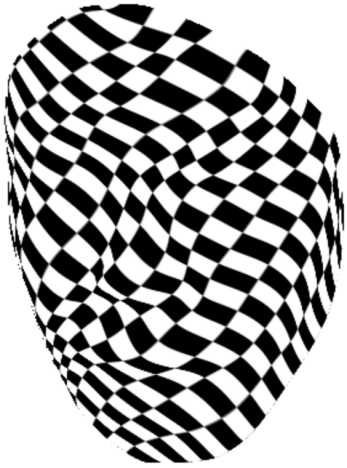
等参线绘制需要计算 B 函数，分别是 n、m 向量方向上的 B 函数取值。然后再根据公式绘制出曲线即可。

```
void calc_B(GLfloat u,int id){
    rep(i,0,n[id]-1){
        if(u != 1 && u >= knt[id][i] && u < knt[id][i+1]) B[id][i][1] = 1;
        else if(u == 1 && u > knt[id][i] && u <= knt[id][i+1]) B[id][i][1] = 1;
        else B[id][i][1] = 0;
    }
    rep(k,2,K[id]+1){
        rep(i,0,n[id]-1){
            B[id][i][k] = 0;
            if(sign(knt[id][i+k-1]-knt[id][i]) != 0)
                B[id][i][k] += (B[id][i][k-1]*(u-knt[id][i]))/(knt[id][i+k-1]-knt[id][i]);
            if(sign(knt[id][i+k]-knt[id][i+1]) != 0)
                B[id][i][k] += (B[id][i+1][k-1])*(knt[id][i+k]-u)/(knt[id][i+k]-knt[id][i+1]);
        }
    }
}
```

六、设置曲面纹理

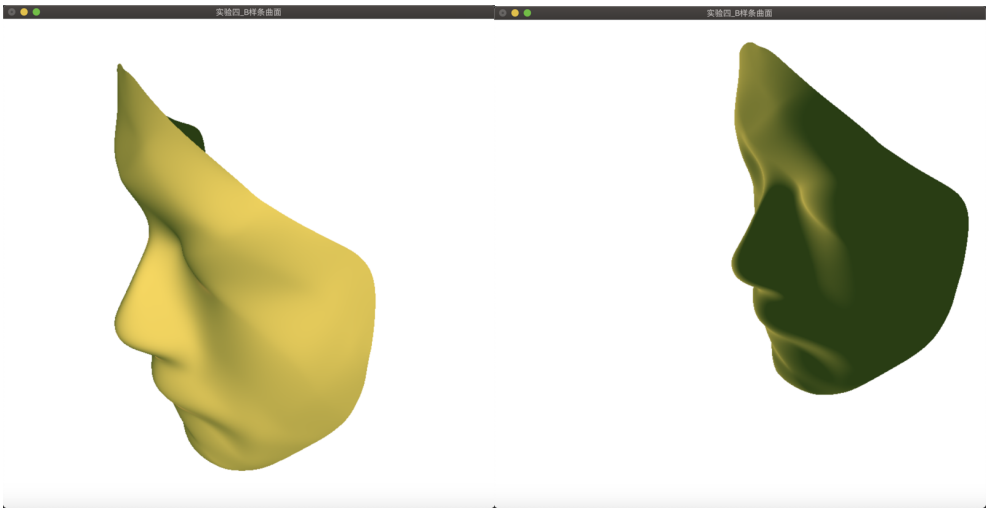
由于没有现成的纹理进行贴图，因此此处自己设置了纹理坐标，然后调用了 API 进行纹理贴图。

```
void show_texture(){
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glRotatef(1.0, 0.7, -0.6, 1.0); // 旋转变换
    glEnable(GL_TEXTURE_2D);
    glBindTexture(GL_TEXTURE_2D, texGround);
    glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);

    float uTextureknots[4] = {0.0, 0.0, 12.0, 12.0};
    float vTextureknots[4] = {0.0, 0.0, 7.0, 7.0};
    float texturePoints[2][2][2] =
    {
        {{0.0, 0.0}, {0.0, 5.0}},
        {{5.0, 0.0}, {5.0, 5.0}}
    };

    gluBeginSurface(theNurb); // 开始曲面绘制
    gluNurbsSurface(theNurb, 4, uTextureknots, 4, vTextureknots, 4, 2, texturePoints[0][0], 2, 2, GL_MAP2_TEXTURE_COORD_2);
    gluNurbsSurface(theNurb, n[0]+K[0]+1, knot1, n[1]+K[1]+1, knot2, n[1] * 3, 3, ctlpoints, K[0]+1, K[1]+1, GL_MAP2_VERTEX_3); //
        定义曲面的数学模型，确定其形状
    gluEndSurface(theNurb); // 结束曲面绘制
    glDisable(GL_TEXTURE_2D);
}
```



七、加入旋转、平移、放大、缩小、光照

旋转、平移、放大、缩小、光照均从上一个实验中复用即可。

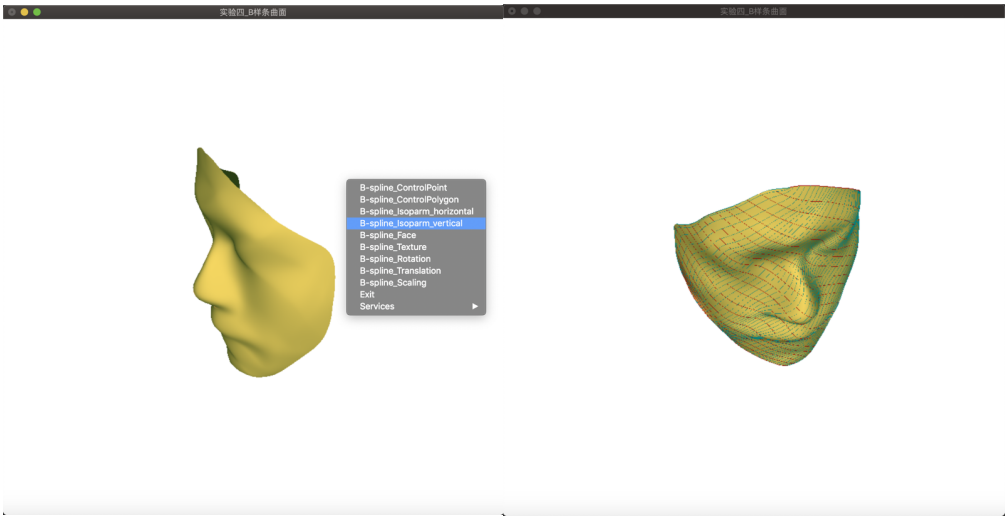


八、交互设计

本实验一共有九个功能。分别是显示控制点、显示控制多边形、显示等参线、显示面、显示贴图、放大（缩小）、旋转、移动、光照。

其中显示等参线部分分为四个功能，分别是显示 u 方向上的等参线、v 方向上的等参线，以及扩大等参线密度、缩小等参线密度。

其中大部分功能都定义在了手表右键菜单中。只有光照定义在了键盘的“1”键上，u 方向等参线扩大密度为“q”，缩小密度为“w”，v 方向等参线扩大密度为“a”，缩小密度为“s”。



实验总结：

1. 实验中画出曲面与曲面贴图为 API 函数调用，填入指定参数即可完成。曲面控制点、曲面控制多边形、曲面等参线均为手动实现。

2. 其中曲面贴图部分，采用了实验三中对于 bmp 文件的读取并自己手动设置了纹理坐标实现了贴图。

3. 其中曲面等参线部分，一开始左右端点取值错误导致找了很久的问题。主要问题出在计算 B(i,1)(u)过程中，ui <= x < u(i+1)时，B(i,1)(u)=1。而不是 ui < x < u(i+1)时，B(i,1)(u)=1。需要进行一定的修改即可正确画出等参线。

$$B_{i,1}(u) = \begin{cases} 1 & u_i < x < u_{i+1} \\ 0 & Otherwise \end{cases}$$

$$B_{i,k}(u) = \frac{u-u_i}{u_{i+k-1}-u_i} B_{i,k-1}(u) + \frac{u_{i+k}-u}{u_{i+k}-u_{i+1}} B_{i+1,k-1}(u)$$

4. 此实验一共有 9 个功能，分别是显示控制点、显示控制多边形、显示等参线、显示面、显示纹理、放大（缩小）、旋转、移动、光照。其中显示等参线部分包括 u、v 两个方向上的等参线，以及两个方向等参线的扩大（缩小）密度。

   其中定义在右键菜单上的功能有（从上往下）：显示控制点、显示控制多边形、显示 u 方向等参线、显示 v 方向等参线、显示面、显示纹理、旋转、平移、放大（缩小）。

   其中定义在键盘上的功能有光照（“1”键控制）、扩大 u 方向等参线密度（“q”）、缩小 u 方向等参线密度（“w”）、扩大 v 方向等参线密度（“a”）、缩小 v 方向等参线密度（“s”）。