几何造型大作业《球面与球面求交》实验报告

孙梓健　2022213885　软硕 222 班

1　实验目的

- 掌握球面与球面的求交算法
- 掌握球面、圆等形状在三维空间中的方程，以及球面几何中圆的参数曲线表达
- 熟悉 Qt 编程的基本思路和方法
- 熟悉 OpenGL 项目的结构和编程方法

2　项目环境

- Windows 11 22H2
- Qt 6.4.2
- OpenGL 4.6.0

3　实验内容

3.1 几何形状数据结构定义

3.1.1　球面类

球面类 Sphere 定义了球面的几何参数：球心、半径，同时定义了获取球面上任意点的 UV 坐标和任意圆的参数曲线的函数，分别在 3.4.1 小节和 3.4.2 的(2)小节详细描述。

3.1.2　三维圆类

三维圆类 Circle3D 用于表示交线圆，定义了圆的几何参数：圆心、半径，以及圆的局部空间坐标系，该坐标系以圆所在平面的法向为z轴，其x轴和y轴的定义在 3.4.2 的(1)小节讲述。

3.2 OpenGL 组件实现

3.2.1　相机模型

OpenGL 的相机由三个参数向量控制，组成视图变换矩阵 ViewMatrix：

- eye：位置向量，相机在世界坐标系中的位置
- front：方向向量，相机指向镜头对准的物体的方向
- up：相机向上的方向在世界坐标系中的方向

此外，还有两个额外的参数，用于控制视角变换：

- 投影变换矩阵 ProjectionMatrix，用于将三维坐标映射到二维窗口
- 视角范围 view_size：表示窗口能展示的空间范围

```
// 相机
QVector3D cam_pos;
QVector3D cam_front;
QVector3D cam_up;
QMatrix4x4 cam_proj_mat;
QMatrix4x4 cam_view_mat;
float view_size;
```

3.2.2 鼠标事件响应

本程序设计了鼠标控制视角移动、旋转和放缩三种操作的响应，分别如下所述：

(1) 移动

A. 根据鼠标移动量计算纵向和横向偏移

B. 分别乘到相机的右方向和上方向的单位向量，得到三维坐标偏移

C. 调整视图变换矩阵

(2) 旋转

A. 根据鼠标移动量计算仰角 yaw 和偏向角 pitch

B. 将角度投影到相机方向向量

C. 调整视图变换矩阵

(3) 放缩

A. 获取滚轮滚动值

B. 计算新的视角范围

C. 调整投影变换矩阵

3.2.3 着色器绑定

定义了着色器必需的两个参数：VBO 和 VAO。VBO 是顶点缓冲对象，主要作用是一次性将一大批顶点数据发送到 GPU 上；VAO 是顶点数组对象，记录一次绘制所需要的所有 VBO 的信息，比如顶点数据的位置、数据的格式、阴影等相关信息，并将其保存在 VBO 的特定位置，绘制的时候就能直接在这个位置调用。除此之外，定义了法向缓冲对象 VNO，存储顶点的法向信息。

```
// 着色器
QOpenGLShaderProgram* shader_program;

QOpenGLBuffer* vbo;
QOpenGLVertexArrayObject* vao;
QOpenGLBuffer* vno;
```

着色器文件分为顶点着色器.vert 文件和片段着色器.frag 文件：

- 顶点着色器：对顶点的法向进行计算和变换

```
#version 330 core
layout(location = 0)in vec3 pos;
layout(location = 1)in vec3 nor;
uniform mat4 ProjectionMatrix;
uniform mat4 ViewMatrix;
out vec3 o_nor;
void main(void)
{
    gl_Position = ProjectionMatrix * ViewMatrix * vec4(pos, 1.0);
    vec4 tmp_nor = ProjectionMatrix * ViewMatrix * vec4(nor, 1.0);
    o_nor = normalize(tmp_nor).xyz;
    o_nor = nor;
}
```

- 片段着色器：接收顶点着色器的输出，计算屏幕上每个像素的颜色值

```
#version 330 core
uniform vec4 ColAttr;
uniform bool UseNor;
in vec3 o_nor;
void main() {
    if (UseNor) {
        float coff = dot(o_nor, vec3(0.9, 0.9, 0.9));
        coff = (coff+2.0)/2.0;
        gl_FragColor = vec4(ColAttr.xyz * coff, ColAttr.w);
    }
    else {
        gl_FragColor = ColAttr;
    }
}
```

3.2.4 绘制圆

绘制圆的基本思路是从圆上的某一个点出发，以法向为轴旋转一周绘制离散点，只要步长足够小，绘制出来的圆就足够光滑。具体步骤如下：

(1) 计算经过圆心且与法向垂直的向量 dir，具体计算方法与 3.4.2 的(1)小节计算局部坐标系的x轴的方法相同，此处暂不赘述；

(2) 定义分辨率为 180，即 2°为一步长；

(3) 以圆心为端点，向 dir 指向的方向走圆的半径长，得到起点，从起点出发，每次逆时针旋转 2°得到下一个点；

(4) 绘制所有点。

3.2.5 绘制球面

绘制球面采用三角网格法，构建球面的 mesh 网格，将每个面片绘制出来即得到球面，与画圆同理，只要面片足够小，绘制出来的球面就足够光滑。具体步骤如下[1]：

(1) 定义分辨率为 100，即将球分为纵向 100 层，横向 200 层；

(2) 按照分辨率采样计算球面上的所有离散点及其法向（法向为球心指向离散点的单位向量）；

(3) 除最高点和最低点形成的盖状结构所包含的点外，每个点参与了周围六个三角面片的组成，如图 1 所示，分别计算每个三角面片的法向（为三个顶点法向的均值），并将顶点保存到 vertices 数组，法向保存到 normals 数组；

$$\vec{n} = \frac{1}{3}(\overrightarrow{n_1} + \overrightarrow{n_2} + \overrightarrow{n_3})$$

(4) 最高点和最低点参与了 200 个三角面片的组成，如图 2 所示，按照(3)的方法计算这些三角面片的顶点和法向，分别保存到 vertices 和 normals 中；

(5) 绘制所有三角面片。

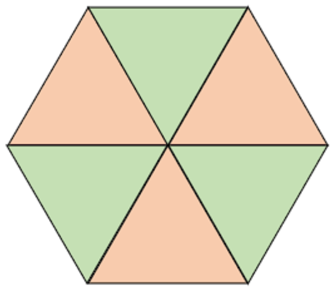

图 1　三角面片局部示意图

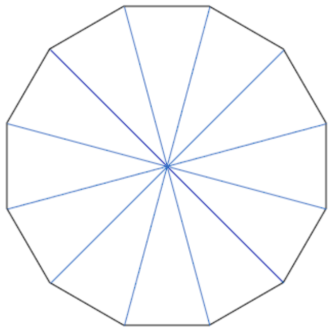

图 2　最高点和最低点示意图

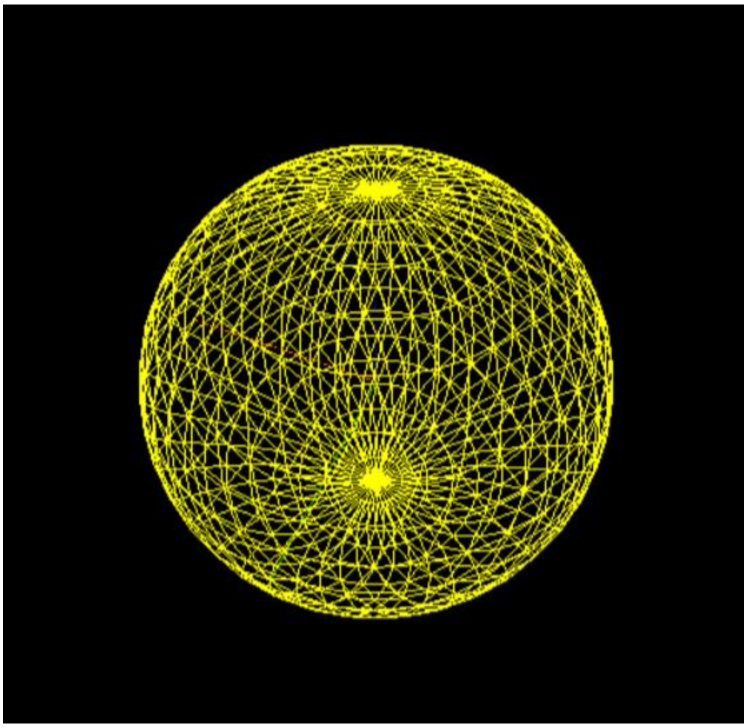

图 3　完整的球面网格示意图

3.3 球面求交

本实验中，支持的两个球面的空间位置关系有 6 种，分别为：不相交、外切、相交、内切、包含、重合，其中不相交和包含看作一种。重合本质上算作内切的一种，但是其相交部分为整个球面，与内切相交于一点的情况并不相同，所以分开处理。

求交函数定义如下所示：

```
/**
 * @param   res   求交结果参数
 * @param   s1    球 1 的指针
 * @param   s2    球 2 的指针
 * @return  求交结果类型(0-不相交 1-有一个交点 2-有一个交线圆 3-重合)
 */
int intersectSphereSphere(vector<float>& res, const Sphere* s1, const Sphere* s2)
{
    QVector3D c1 = s1->center();
    float r1 = s1->radius();
    QVector3D c2 = s2->center();
    float r2 = s2->radius();

    float d = (c1 - c2).length();
    if (isZero(d) && floatCmp(r1, r2) == 0) {
        return 3;
    }

    if (floatCmp(d, r1 + r2) > 0) {
        return 0;
    }
    else if (floatCmp(d, r1 + r2) == 0) {
        QVector3D n = c2 - c1;
        n.normalize();

        QVector3D p1 = c1 + r1 * n;
        QVector3D p2 = c2 - r2 * n;
        QVector3D p = (p1 + p2) / 2;

        res.push_back(p.x());
        res.push_back(p.y());
        res.push_back(p.z());
        return 1;
    }
    else if (floatCmp(d, r1 + r2) < 0 && floatCmp(d, fabs(r1 - r2)) > 0) {
        float cosθ1 = (r1 * r1 + d * d - r2 * r2) / (2 * r1 * d);
```

```
        float cosθ2 = (r2 * r2 + d * d - r1 * r1) / (2 * r2 * d);

        float d1 = r1 * cosθ1;
        float d2 = r2 * cosθ2;

        QVector3D n = c2 - c1;
        n.normalize();

        QVector3D p1 = c1 + d1 * n;
        QVector3D p2 = c2 - d2 * n;
        QVector3D p = (p1 + p2) / 2;

        float h = 2 * areaTriangle(r1, r2, d) / d;

        res.push_back(p.x());
        res.push_back(p.y());
        res.push_back(p.z());
        res.push_back(n.x());
        res.push_back(n.y());
        res.push_back(n.z());
        res.push_back(h);

        return 2;
    }
    else if (floatCmp(d, fabs(r1 - r2)) == 0) {
        QVector3D n = c1 - c2;
        if (floatCmp(r1, r2) > 0) {
            n = c2 - c1;
        }
        n.normalize();

        QVector3D p1 = c1 + r1 * n;
        QVector3D p2 = c2 + r2 * n;
        QVector3D p = (p1 + p2) / 2;

        res.push_back(p.x());
        res.push_back(p.y());
        res.push_back(p.z());
        return 1;
    }
    else {
        return 0;
    }
}
```

其中，求交结果类型和求交结果参数是一致的，如果求交结果为 0 或 3，则 res 为空；如果结果为 1，则 res 长度为 3，分别为交点在三维空间的x, y, z坐标；如果结果为 2，则 res 长度为 7，分别为交线圆圆心的x, y, z坐标，交线圆所在平面的法向的$\vec{\imath}, \vec{\jmath}, \vec{k}$系数，以及交线圆的半径。球面求交算法见附录 1。

3.4 求交结果处理

根据求交函数输出结果，需要后续处理的情况分为交点和交线圆两类：

3.4.1 交点

要求输出交点在三维空间中的坐标，以及交点在两球上的UV参数。对于后者，将全局坐标系平移到球心，建立新的局部坐标系，计算交点在局部坐标系中的坐标(x', y', z')，再将局部坐标代入球面的参数方程，计算u和v：

$$\begin{cases} x' = R\cos v\cos u \\ y' = R\cos v\sin u \\ z' = R\sin v \end{cases}$$

$$\Rightarrow \begin{cases} u = \arctan\dfrac{y'}{x'} \\ v = \arcsin\dfrac{z'}{R} \end{cases}$$

在计算u值时，由于规定$u \in [0,2\pi)$，采用`float atan2f(float y, float x)`函数计算位于$[-\pi, \pi)$的方位角，再将负的区间移动到$[\pi, 2\pi)$。具体的函数实现如下：

```
/**
 * @param   p     球面上一点
 * @return  uv 坐标
 */
QVector2D Sphere::getPointUV(const QVector3D& p)
{
    float u = isNonNegative(p.y() - _c.y()) ? atan2f(p.y() - _c.y(), p.x() -
_c.x()) : atan2f(p.y() - _c.y(), p.x() - _c.x()) + 2 * M_PI;
    float v = asinf((p.z() - _c.z()) / _r);

    return QVector2D(u, v);
}
```

3.4.2 交线圆

要求输出交线圆在三维空间中的方程，以及圆在两个球面上的参数曲线。

(1) 圆在三维空间中的方程

通常情况下，圆在三维空间中通过其所在平面的局部坐标系进行表示，局部坐标系的原点位于圆的圆心，z轴为圆所在平面的法向，x轴和y轴为圆所在平面上任意两条相互正交的

单位向量。此时圆方程在局部坐标系中表示为局部二维方程，即：

$$x^2+y^2=r^2$$

计算局部x轴和局部y轴，只需利用向量叉乘运算的特点，两个向量叉乘的结果不为零，则所得向量总是垂直于原来的这两个向量。令局部x轴和局部y轴的向量分别为$\vec{a}$和$\vec{b}$，圆的法向为$\vec{n}$，将$\vec{n}$叉乘$\vec{i}$，若结果不为零，那么它必然垂直于$\vec{n}$，将它单位化后就得到$\vec{a}$；如果叉乘结果为零，再用$\vec{n}$叉乘$\vec{j}$或$\vec{k}$，将结果单位化得到$\vec{a}$。得到$\vec{a}$后，将$\vec{n}$叉乘$\vec{a}$则得到$\vec{b}$。具体函数实现如下：

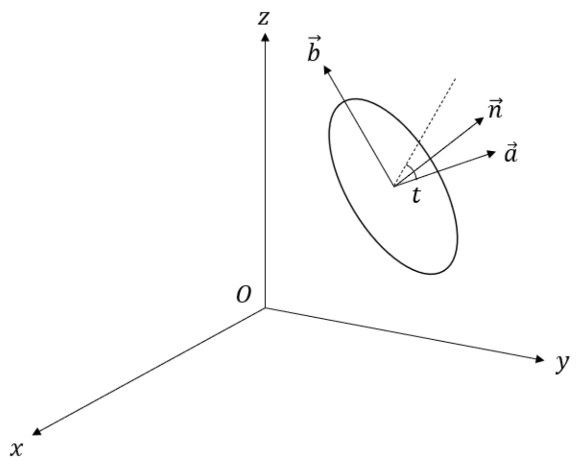

图 4　圆的局部坐标系

```
/**
 * @param   p     局部坐标系的 x 轴向量
 * @param   q     局部坐标系的 y 轴向量
 * @return  无
 */
void Circle3D::getLocalCoor(QVector3D& p, QVector3D& q) const
{
    QVector3D i(1, 0, 0);
    p = QVector3D::crossProduct(_n, i).normalized();
    if (p.isNull()) {
        QVector3D j(0, 1, 0);
        p = QVector3D::crossProduct(_n, j).normalized();
    }
    q = QVector3D::crossProduct(_n, p).normalized();
}
```

此外，在三维空间中圆可以表示为参数方程。对于一个以(c_1,c_2,c_3)为圆心，向量$\vec{n}$为法向，半径为r的圆，它的参数方程为[2]：

$$\begin{cases}x(t)=c_1+ra_1\cos t+rb_1\sin t\\ y(t)=c_2+ra_2\cos t+rb_2\sin t\\ z(t)=c_3+ra_3\cos t+rb_3\sin t\end{cases}$$

其中，(a_1,a_2,a_3)和(b_1,b_2,b_3)分别对应局部坐标系的单位向量$\vec{a}$和$\vec{b}$，t表示圆上一点相对于圆心的旋转角。具体推导过程见附录 2。

(2) 圆在球面上的参数曲线

在球面几何中，圆在球面上的参数曲线为[3]：

$$s(u,v) = kR\cos v\sin(u-\gamma) - R\sin v + d\sqrt{1+k^2} = 0$$

此公式可以表达不与球的赤道平面相垂直的各个大小圆，其中各参数的含义如下所示：

u：圆上一点在球面上的参数u坐标

v：圆上一点在球面上的参数v坐标

R：球的半径

k：圆的斜率，指圆所在平面与球的赤道平面的夹角的正切值

d：圆心到球心的有向距离，当圆心位于赤道平面下方时为负值，范围为$(-R,R)$

γ：初相，指球面上与交线圆平行的大圆与赤道平面的交点的方位角，范围为$[0,\pi)$

而与赤道平面垂直的交线圆，其斜率k无穷大，参数曲线方程表示为：

$$R\cos v\sin(u-\gamma) + d = 0$$

关于球面圆方程的推导见附录 3。

基于以上公式，定义获取球面上一圆的参数的函数如下：

```
/**
* @param   circ     三维圆
* @param   k        圆的斜率
* @param   γ        圆的初相
* @param   d        圆心到球心的距离
* @return  圆是否垂直于赤道平面
*/
bool Sphere::getCircleUV(const Circle3D& circ, float& k, float& γ, float& d)
{
    bool is_vertical = false;
    // 交线圆所在平面与赤道平面夹角正切
    QVector3D n = circ.center() - _c;
    if (isZero(n.z())) {
        is_vertical = true;
    }
    k = tanf(acosf(n.normalized().z()));

    // 大圆与赤道平面交点的方位角
    if (isZero(n.y())) {
        γ = M_PI_2;
    }
    else {
        γ = atan(-n.x() / n.y());
        if (isNegative(γ)) {
            γ += M_PI_2;
        }
```

```
    }

    // 交线圆与球心的距离
    d = n.length();
    if (floatCmp(circ.center().z(), _c.z()) < 0) {
        d = -d;
    }

    return is_vertical;
}
```

其中，三个参数的计算方式如下所示：

(1) 斜率k：圆的法向与z轴坐标向量的夹角θ的正切

$$k = \tan\arccos\frac{\vec{n}\cdot\vec{k}}{|\vec{n}|\cdot|\vec{k}|} = \tan\arccos n_z$$

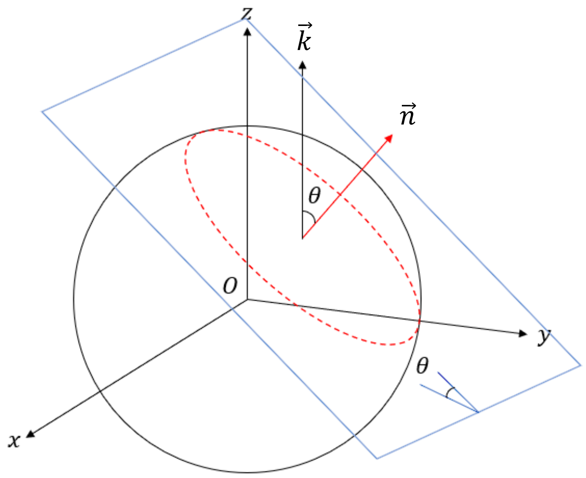

图 5　圆的斜率示意图

(2) 初相γ：

考虑以球心为原点的局部坐标系，取与交线圆平行的大圆所在平面$Ax + By + Cz + D = 0$，其中$A = n_x, B = n_y, C = n_z, D = 0$

令$z = 0$，得到大圆与赤道平面的交线$n_x x + n_y y = 0$，交线的倾斜角即为初相，若$n_y = 0$，则交线与y轴重合，此时有：

$$\gamma = \frac{\pi}{2}$$

否则，

$$\gamma = \arctan\left(-\frac{n_x}{n_y}\right)$$

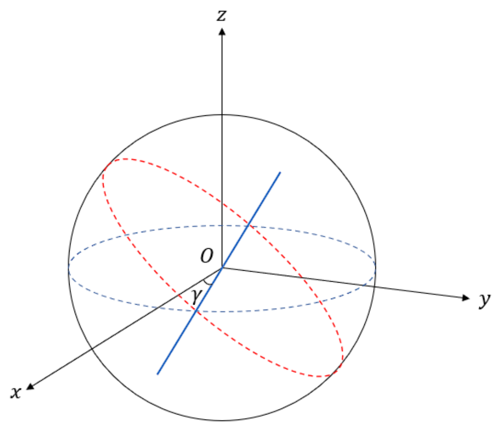

图 6　圆的初相示意图

(3) 距离d：圆心到球心的有向距离

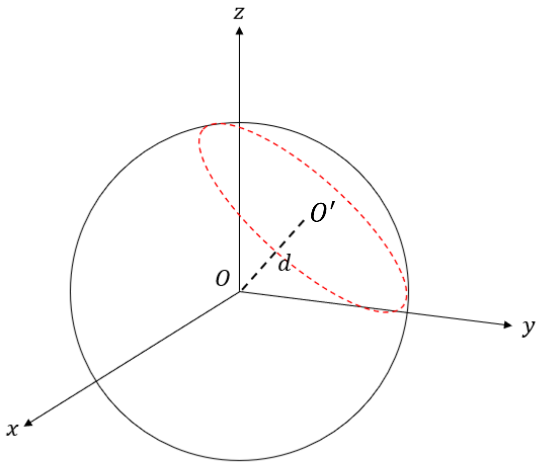

图 7　圆心到球心距离示意图

4　程序使用说明

本程序主界面如图 8 所示：

图 8　圆心到球心距离示意图

支持“绘制球”、“求交”、“清除”三个操作，在左上角工具栏中显示。单击“绘制球”，弹出输入参数对话框，如图 9 所示，需要输入球面的球心坐标和半径，四个参数都是必填项，其中半径要求为大于 0 的实数。

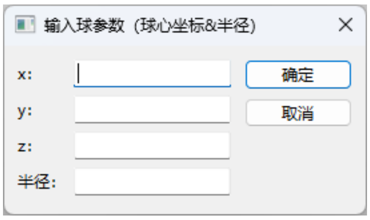

图 9　输入球面参数对话框

输入正确的参数后，在图形窗口将会显示球面，如图 10 所示。

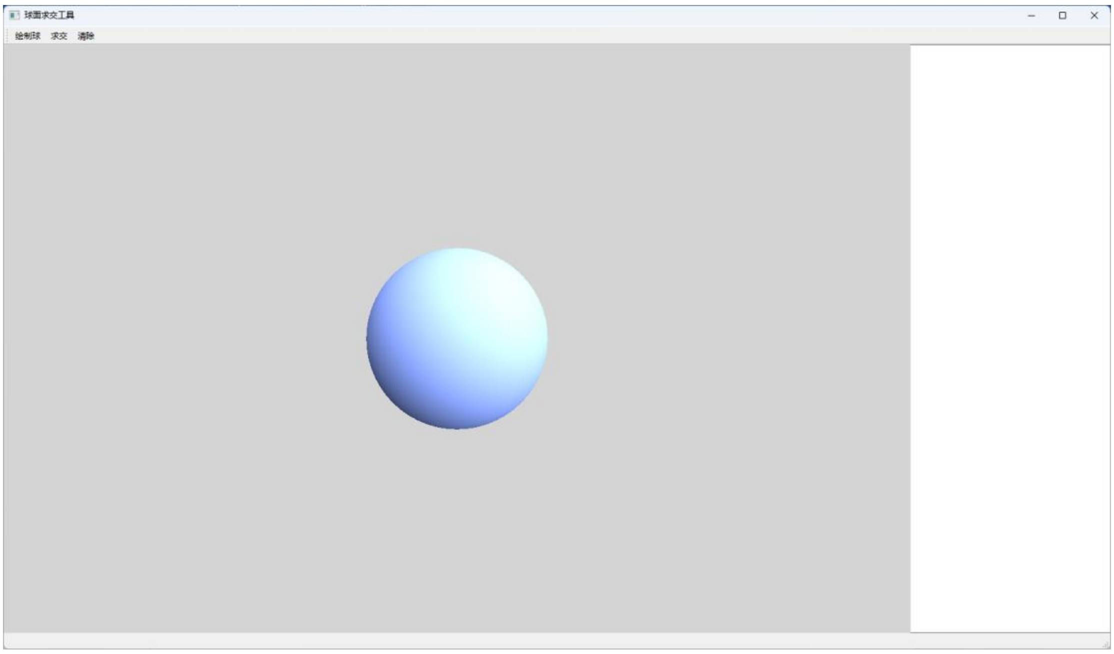

图 10　绘制球面

需要绘制两个球面，才能完成求交操作。点击“求交”，在图形界面中将会显示交点或

交线（黑色），在右侧边栏中输出原始球面和交点或交线的相关参数，如图 11 所示。

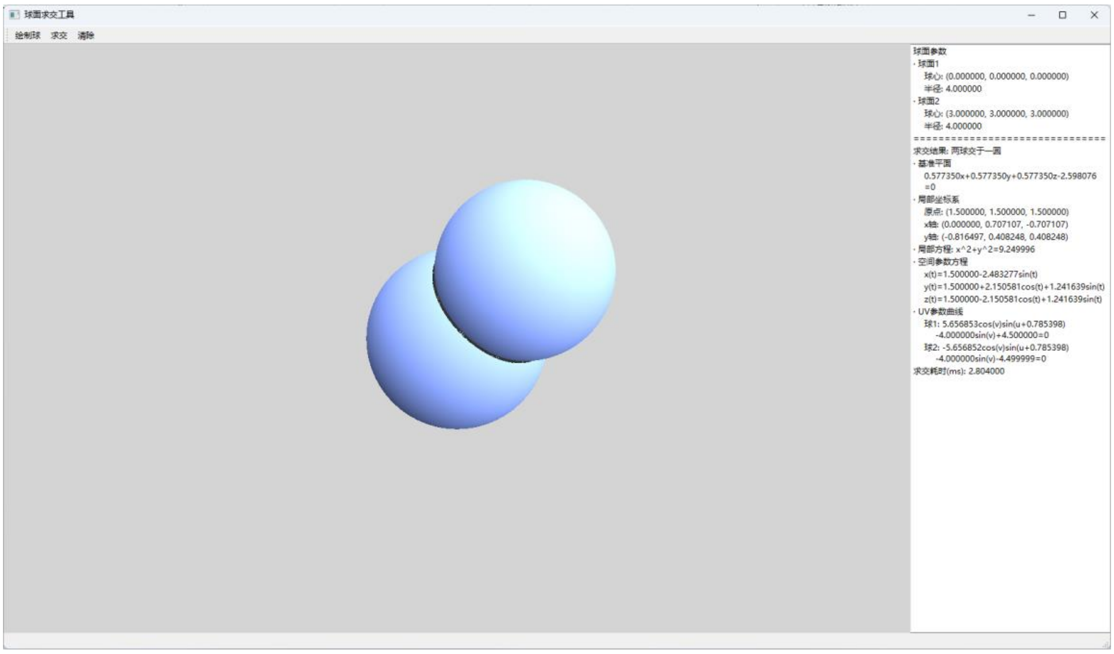

图 11　求交结果展示

点击“清除”按钮，图形界面和参数栏将会清空，并支持绘制新的球面和进行下一次求交。

附录 1　球面求交算法

三维空间中两个球面的空间位置关系由两球半径R_1, R_2与两球球心之间的距离d的大小关系所决定，共分为五种情况（以下图示中左侧为截面示意图，两球球心的连线为蓝色，相交部分用红色表示）：

1. $R_1 + R_2 < d$，两球不相交

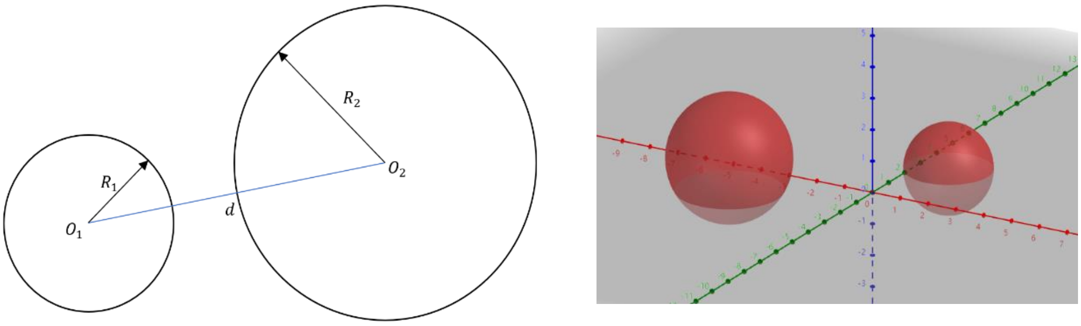

图 6　两球不相交

2. $R_1 + R_2 = d$，两球外切，此时，两球相切于点$O_1 + R_1 * \frac{\overrightarrow{O_1O_2}}{d}$

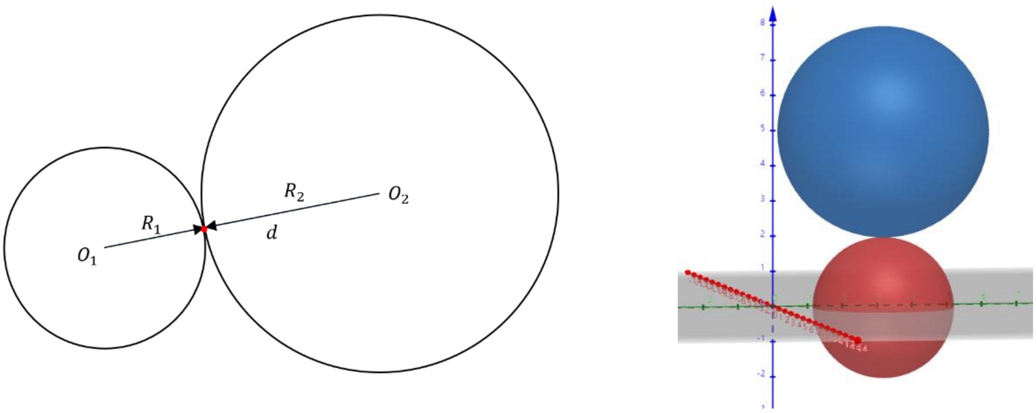

图 6　两球外切

3. $|R_1 - R_2| < d < R_1 + R_2$，两球相交于一个交线圆，需要求出交线圆的圆心、法向和半径

令交线圆圆心为P，交线圆上任取一点为Q，在ΔO_1O_2Q中，通过余弦定理计算$\angle QO_1O_2$

$$\cos\angle QO_1O_2 = \frac{|O_1Q|^2 + |O_1O_2|^2 - |O_2Q|^2}{2 * |O_1Q| * |O_1O_2|} = \frac{R_1^2 + d^2 - R_2^2}{2dR_1}$$

在ΔO_1PQ中，计算交线圆半径r

$$r = R_1 \sin\angle QO_1O_2$$

显然，交线圆的法向为$\frac{\overrightarrow{O_1O_2}}{d}$，则交线圆圆心$P$可表示为

$$O_1 + R_1 \cos\angle QO_1O_2 * \frac{\overrightarrow{O_1O_2}}{d}$$

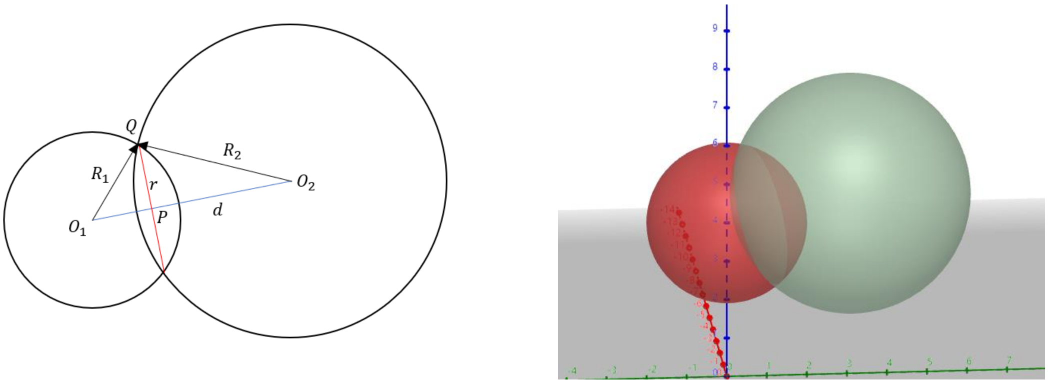

图 6　两球相交

4. $|R_1 - R_2| = d$，两球内切，此时，两球相切于点$O_1 - R_1 * \frac{|\overrightarrow{O_1O_2}|}{d}$（规定下标为 1 的球半径更小）；特别地，如果两球半径相等，则两球完全重合

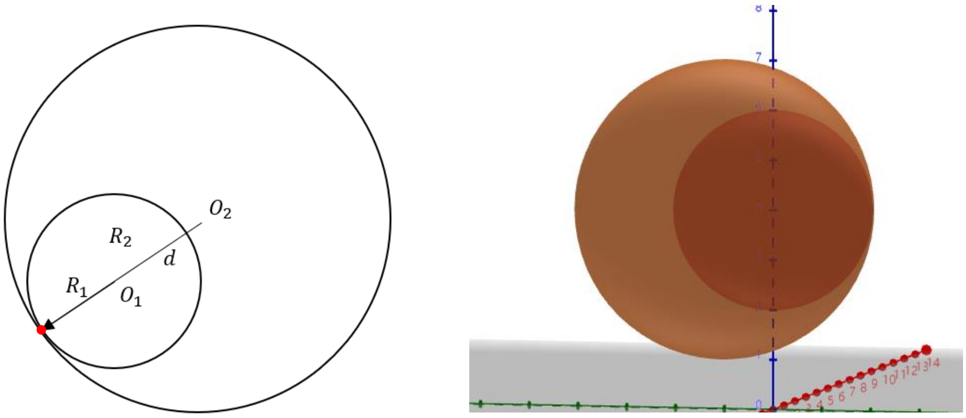

图 6　两球内切

5. $0 \le d < |R_1 - R_2|$，两球不相交，属于包含关系

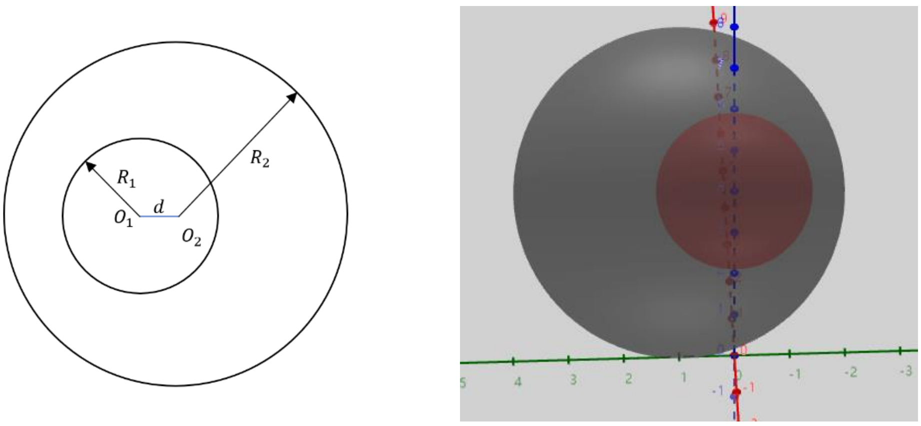

图 6　两球包含

附录 2　交线圆在三维空间中的参数方程

考虑向量形式，如果令交线圆圆心为C，圆上任意一点为P，则P可以表示为：

$$\overrightarrow{OP} = \overrightarrow{OC} + \overrightarrow{CP}$$

在交线圆的局部坐标系中，由向量的平行四边形法则可知，$\overrightarrow{CP}$是坐标向量$\vec{a},\vec{b},\vec{n}$的线性组合，基于几何关系可得

$$\overrightarrow{CP} = p\vec{a} + q\vec{b} + 0\cdot\vec{n} = r\vec{a}\cos t + r\vec{b}\sin t$$

于是有

$$\overrightarrow{OP} = \overrightarrow{OC} + r\vec{a}\cos t + r\vec{b}\sin t$$

展开其坐标分量，得

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} + r\cos t \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} + r\sin t \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

即

$$\begin{cases} x(t) = c_1 + ra_1\cos t + rb_1\sin t \\ y(t) = c_2 + ra_2\cos t + rb_2\sin t \\ z(t) = c_3 + ra_3\cos t + rb_3\sin t \end{cases}$$

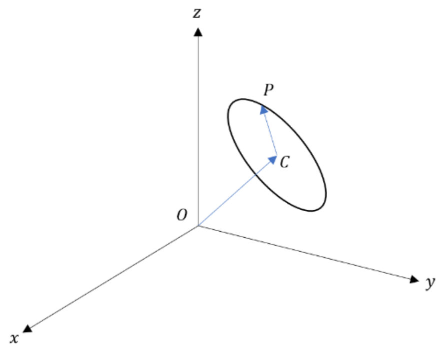

图 6　交线圆上一点的向量表示

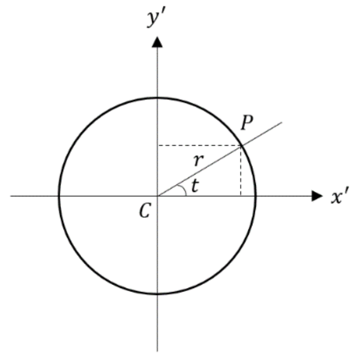

图 6　交线圆局部坐标示意图

附录 3　交线圆在球面上的参数曲线

球面上一圆分为大圆和小圆，大圆是指半径最大的圆，其所在平面经过球心，而其他的圆都是小圆。我们从最简单的情况开始来推导交线圆的参数曲线，即首先探究初相$\gamma = 0$且斜率$k < \infty$的大圆在球面上的公式。

当交线圆为大圆，且初相$\gamma = 0$时，该圆可以看作由球面与其赤道平面的交线圆以x轴为旋转轴旋转一定角度形成，其与赤道平面的交点必然落在x轴上，故初相为 0。令旋转后大圆所在平面与赤道平面形成的角度为$\delta(\delta < \frac{\pi}{2})$，交线圆上任意一点为点$P$，点$P$在赤道平面上的投影为点$A$，点$A$在$x$轴上的投影为点$B$，则有$u = \angle AOB$，$v = \angle POA$，于是有

$$\sin u = \frac{|AB|}{|OA|}, \tan v = \frac{|AP|}{|OA|}.$$

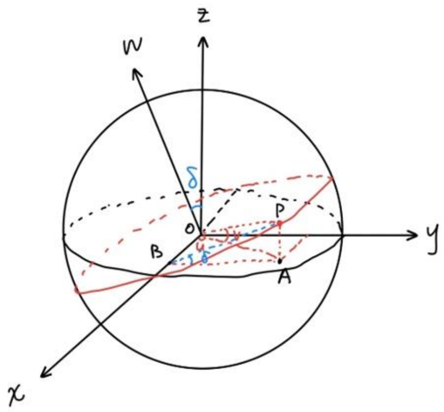

图 6　初相为 0 的大圆

连接BP，在$\mathrm{Rt}\Delta ABP$中，有$\tan\angle PBA = \frac{|AP|}{|AB|} = \frac{\tan v}{\sin u}$，而显然有$BP \perp x$，$AB \perp x$，满足二面角定义，即$\delta = \angle PBA$，于是有$\tan v = \tan\delta \sin u$。按照圆的斜率的定义，有$k = \tan\delta$，于是得到球面上初相为 0 的大圆方程：

$$k\sin u - \tan v = 0.$$

接下来我们就能得到与这个大圆相平行的一系列小圆的方程。已知$k = \tan\delta$，那么有

$$\begin{cases} \sin\delta = \dfrac{k}{\sqrt{1+k^2}} \\ \cos\delta = \dfrac{1}{\sqrt{1+k^2}} \end{cases} \cdots\cdots ①$$

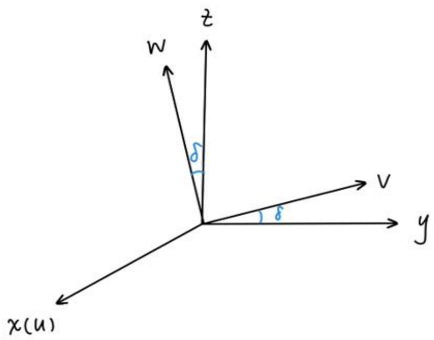

图 6　大圆的局部坐标系

考虑大圆的局部坐标系uvw，其中u轴与原x轴重合，v轴和w轴分别与原y轴和z轴形成角度δ，则有旋转矩阵$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\delta & -\sin\delta \\ 0 & \sin\delta & \cos\delta \end{bmatrix}$。

小圆半径$r = \sqrt{R^2 - d^2}$，那么小圆在uvw空间下的参数方程为

$$\begin{cases} u(\theta) = \sqrt{R^2 - d^2}\cos\theta \\ v(\theta) = \sqrt{R^2 - d^2}\sin\theta \\ w(\theta) = d \end{cases} \cdots\cdots ②$$

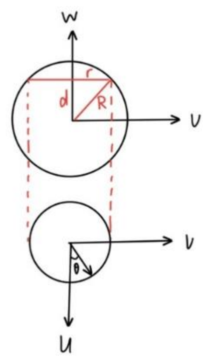

图 6　小圆在uvw空间中的示意图

将uvw空间变换回xyz空间，有如下变换：

$$\begin{bmatrix} x(\theta) \\ y(\theta) \\ z(\theta) \end{bmatrix} = M \begin{bmatrix} u(\theta) \\ v(\theta) \\ w(\theta) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\delta & -\sin\delta \\ 0 & \sin\delta & \cos\delta \end{bmatrix} \begin{bmatrix} u(\theta) \\ v(\theta) \\ w(\theta) \end{bmatrix}$$

代入②式和①式，得

$$\begin{cases} x(\theta) = u(\theta) = \sqrt{R^2 - d^2}\cos\theta \\ y(\theta) = v(\theta)\cos\delta - w(\theta)\sin\delta = \dfrac{\sqrt{R^2 - d^2}}{\sqrt{1 + k^2}}\sin\theta - \dfrac{kd}{\sqrt{1 + k^2}} \\ z(\theta) = v(\theta)\sin\delta + w(\theta)\cos\delta = \dfrac{k\sqrt{R^2 - d^2}}{\sqrt{1 + k^2}}\sin\theta + \dfrac{d}{\sqrt{1 + k^2}} \end{cases}$$

消去参数θ，得

$$\begin{cases} x^2 + \dfrac{(y + kz)^2}{1 + k^2} = R^2 - d^2 \dots\dots ③ \\ z = ky + d\sqrt{1 + k^2} \dots\dots ④ \end{cases}$$

将球面参数方程代入④式，得

$$kR\cos v\sin u - R\sin v + d\sqrt{1 + k^2} = 0$$

从几何视角出发也能得到相同的结论，对球面大圆方程进行变换

$k\sin u - \tan v = 0$

$\Rightarrow kR\cos v\sin u - R\sin v = 0$

$\Rightarrow ky - z = 0$

该方程在球面几何中表示球面直线方程（大圆），在三维空间中表示大圆所在的平面方程，由此想到运用小圆所在的平面方程$ky - z + C = 0$。根据平面距离公式，得到大小圆所在的平面距离$d = \frac{C}{\sqrt{k^2+1}}$，即$C = d\sqrt{1 + k^2}$，则小圆所在平面方程为$ky - z + d\sqrt{1 + k^2} = 0$，代入球面参数方程，得

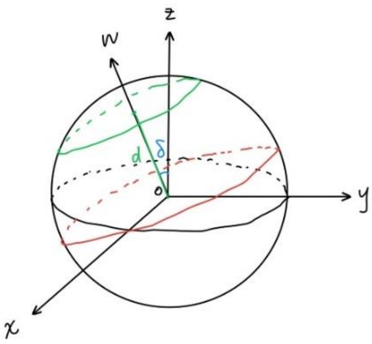

图 6　大小圆所在平面

$$kR\cos v\sin u - R\sin v + d\sqrt{1 + k^2} = 0$$

有了初相为 0 的小圆方程，就可以推导到初相不为 0 的情况。我们知道初相γ的定义是与交线圆平行的大圆与赤道平面的交点的方位角，那么初相不为 0 就意味着大圆以z轴为旋转轴逆时针旋转了γ。

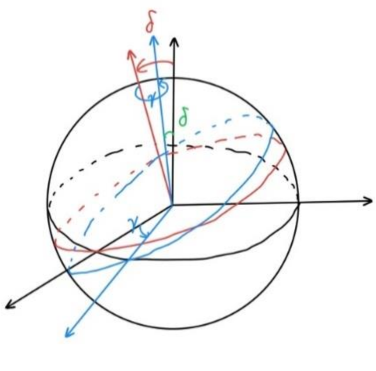

图 6　大圆绕z轴旋转γ

所以，在新的uvw空间中，上述公式均满足，只是和原空间相差了相位γ，则可以直接写出带初相的大圆和小

圆方程：

a) 球面大圆方程：$k\sin(u-\gamma)-\tan v=0$

b) 球面任意圆方程：$kR\cos v\sin(u-\gamma)-R\sin v+d\sqrt{1+k^2}=0$

鉴于笔者水平有限，无法对这个推论给出严格证明，但可以做一些形式上的验证，以此证明逻辑是自洽的，以下给出验证过程。

我们认为，带初相的情况是大圆先绕x轴旋转δ，再绕z轴旋转γ，那么相应的旋转矩阵为

$$M=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\delta & -\sin\delta\\ 0 & \sin\delta & \cos\delta\end{bmatrix}\begin{bmatrix}\cos\gamma & \sin\gamma & 0\\ -\sin\gamma & \cos\gamma & 0\\ 0 & 0 & 1\end{bmatrix}$$

$$=\begin{bmatrix}\cos\gamma & \sin\gamma & 0\\ -\cos\delta\sin\gamma & \cos\delta\cos\gamma & -\sin\delta\\ -\sin\delta\sin\gamma & \sin\delta\cos\gamma & \cos\delta\end{bmatrix}$$

代入坐标变换$\begin{bmatrix}x(\theta)\\ y(\theta)\\ z(\theta)\end{bmatrix}=M\begin{bmatrix}u(\theta)\\ v(\theta)\\ w(\theta)\end{bmatrix}$，得$\begin{cases}x=u\cos\gamma+v\sin\gamma\\ y=-u\cos\delta\sin\gamma+v\cos\delta\cos\gamma-w\sin\delta\\ z=-u\sin\delta\sin\gamma+v\sin\delta\cos\gamma+w\cos\delta\end{cases}$ ……⑤

此时，小圆的参数方程和球面参数方程都需要相应地带上初相γ，即

$$\begin{cases}u(\theta)=\sqrt{R^2-d^2}\cos(\theta-\gamma)\\ v(\theta)=\sqrt{R^2-d^2}\sin(\theta-\gamma)\\ w(\theta)=d\end{cases}\ \cdots\cdots⑥$$

$$\begin{cases}x=R\cos v\cos(u-\gamma)\\ y=R\cos v\sin(u-\gamma)\\ z=R\sin v\end{cases}\ \cdots\cdots⑦$$

将⑥式代入⑤，再运用积化和差公式，得

$$\begin{cases}x=\sqrt{R^2-d^2}\cos(\theta-\gamma)\\ y=\dfrac{\sqrt{R^2-d^2}}{\sqrt{1+k^2}}\sin(\theta-\gamma)-\dfrac{kd}{\sqrt{1+k^2}}\\ z=\dfrac{k\sqrt{R^2-d^2}}{\sqrt{1+k^2}}\sin(\theta-\gamma)+\dfrac{d}{\sqrt{1+k^2}}\end{cases}$$

消去参数θ，得$z=ky+d\sqrt{1+k^2}$

再将⑦式代入，得$kR\cos v\sin(u-\gamma)-R\sin v+d\sqrt{1+k^2}=0$，得到了和初相为 0 时相同的式子。

最后，我们考虑圆的斜率无限大的情况。依然优先考虑最简单的情况，即位于xOz平面内的大圆和与之平行的一系列小圆，不难发现，圆上所有的点的y坐标绝对值始终等于d。我们规定圆心在y轴正向时的$d<0$，则有

$$R\cos v\sin u+d=0$$

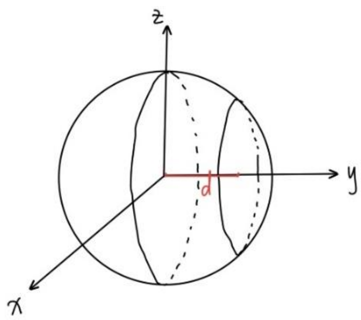

图 6 xOz平面内的大圆和平行的一系列小圆

对于不平行于xOz平面的圆，都可以看作带有初相γ的大圆平移所得，同样按照上面的旋转规则，在uvw空间内圆方程满足$R\cos v\sin u+d=0$，则在原xyz空间中方程为

$$R\cos v\sin(u-\gamma)+d=0$$

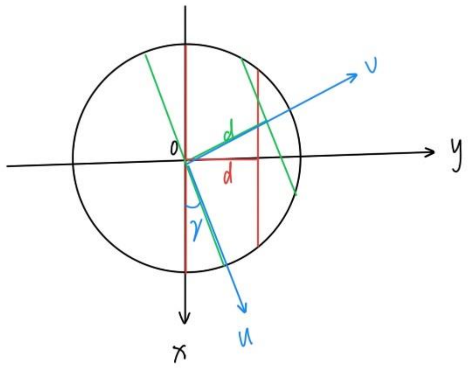

图 6　斜率无限大且带初相的圆

参考资料

[1] 球面三角网格绘制算法 https://blog.csdn.net/u014132143/article/details/111761448

[2] 三维空间中圆的参数方程 https://blog.sina.com.cn/s/blog_6496e38e0102vi7e.html

[3] 球面圆的方程的推导过程 https://zhuanlan.zhihu.com/p/113501566