Computação Gráfica 3ª Parte

Trabalho realizado por:

João Figueiredo Martins Peixe dos Santos

Francisco Alves Andrade

Luís Filipe Cruz Sobral

Paulo Silva Sousa

A89520 João Santos

A89474 Luís Sobral

A89465 Paulo Sousa

A89513 Francisco Andrade

Grupo 14
Projeto CG
2020/2021
Universidade do Minho

Índice

Índice de Figuras

1 Introdução

A terceira fase do projeto tem como objetivo o desenvolvimento de novas funcionalidades associadas ao Generator e ao Engine desenvolvidos nas fases anteriores.

Para isso, alteramos o código de modo a utilizar VBO's no desenho do sistema solar. Tornamos o nosso sistema solar dinâmico e para tal utilizamos a curva de Catmull-Rom's de modo a definir as trajectorias dos planetas.

Para além disso, o generator é agora capaz de ler um ficheiro com os índices e pontos de controlos dos patches, utilizando curvas de bezier.

2 Arquitetura do Código

Tal como nas duas fases anteriores, continuamos a utilizar duas aplicações: *Generator* e *Engine*. Todas as alterações a que estas aplicações foram sujeitas durante o processo de desenvolvimento serão descridas de seguida:

2.1 Generator

Nesta fase o gerador tem de ser capaz de desenvolver um novo modelo baseado num *Bezier Patch*. Tendo isso em mente, desenvolvemos uma primitiva que calcula os pontos necessários para desenhar o modelo final baseado num ficheiro que possui os pontos de controlo.

2.1.1 Leitura de Ficheiro

O ficheiro que contém a informação para gerar os vértices do modelo Bezier pretendido, está dividido em 4 partes:

- 1ª linha - Número de patches (patches_nr)
- patches_nr linhas seguintes - Patches: cada linha contém 16 pontos de controlo que constitui uma patch.
- Linha seguinte - Número de Pontos de Controlo (points_nr)
- points_nr linhas seguintes - Pontos de Controlo

Tendo em consideração esta organização da informação, começamos por ler a informaçao relativa ao número de patches (patches_nr). De seguida, guardamos para cada patch os índices dos pontos de controlo num array multidimensional. Por fim, guardamos as coordenadas dos pontos de controlo num vector<Shape> patches, que serão posteriormente acedidos para calcular os vértices do modelo que pretendemos representar.

2.1.2 Curvas de Bezier

Uma curva de Bezier nessessita de 4 pontos, denominados de pontos de controlo e estão definidos em XYZ. Após obtermos as coordenadas dos pontos provenientes da leitura do ficheiro vamos calcular, para cada patch, os pontos necessários. Para tal, vamos usar:

- divisions = 1.0 / tessellation
- u=us × divisions, us=[1..tessellation[;
- v=vs × divisions, vs=[1..tessellation[;

Posto isto, iremos percorrer os valores de u e v e vamos utilizar estas fórmulas para calcular os pontos dos triângulos que depois serão escritos no ficheiro de output. Assim, vamos calcular 4 pontos correspondentes a dois triângulos, por cada iteração do ciclo que percorre u e v.

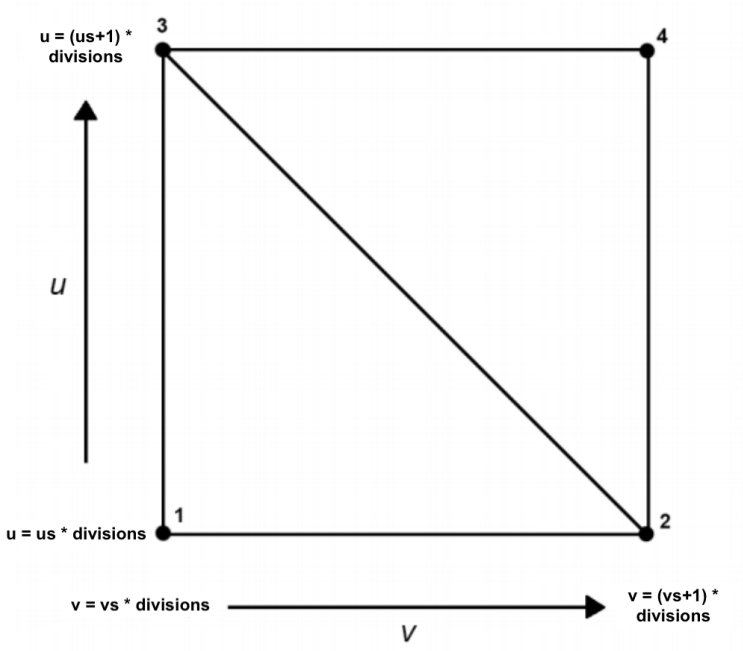

Figure 1: Calculo pontos com base em u e v

Seguindo as fórmulas descritas, iremos calcular as coordenadas de cada um dos pontos, tendo em conta que para cada um deles apenas varia o u e o v. Começamos por escrever no ficheiro output os pontos 1, 2 e 3, correspondentes ao primeiro triângulo, e seguidamente os pontos 3, 2 e 4, que correspondem ao segundo triângulo. Para o calculo de cada pontos usamos os seguintes coeficientes:

- b0 = (1 - u) * (1 - u) * (1 - u)
- b1 = 3 * u * (1 - u) * (1 - u)
- b2 = 3 * u * u * (1 - u)
- b3 = u * u * u
- P(t) = P0*b0 + P1*b1 + P2*b2 + P3*b3

2.2 Engine

2.2.1 Implementação de VBOs

Com o objetivo de aumentar a eficiência do nosso projeto, foi nos pedido que implementassemos VBOs (vertex buffer object).

Para isso, adicionamos duas novas variáveis de instância ao nosso grupo: *buffer* e *buffer_size*.

```
class Group {
private:
    float buffer_size;
    GLuint buffer;
    vector<Transformation*> transf;
    vector<Shape*> models;
    vector<Group*> groups;
    bool primary;
```

Figure 2: Variáveis de instância da classe Group

Após isso, criamos uma função chamada *setBuffer* que inicializa o nosso buffer com os pontos gerados a partir do ficheiro introduzido pelo utilizador.

```
void Group::setBuffer(){
    buffer_size = 0;

    for(int i=0; i<models.size(); i++)
        buffer_size += models[i]->size();

    int index = 0;
    float* points_arr = (float*) malloc( size: sizeof(float) * buffer_size * 3);

    for(int i=0; i<models.size(); i++){
        for(int j=0; j<models[i]->size(); j++) {
            Point *aux = models[i]->getPoint(j);
            points_arr[index] = aux->getX();
            points_arr[index+1] = aux->getY();
            points_arr[index+2] = aux->getZ();
            index+=3;
        }
    }

    glGenBuffers( n: 1, &buffer);
    glBindBuffer(GL_ARRAY_BUFFER, buffer);
    glBufferData(GL_ARRAY_BUFFER,  size: sizeof(float) * buffer_size * 3, points_arr, GL_STATIC_DRAW);

    free(points_arr);
}
```

Figure 3: Função *setBuffer*

Esta função é chamada cada vez que criamos um grupo a partir do parse:

```
if (strcmp(elem->Name(), "group") == 0) {
    sGroup = parseGroup(elem, primary: false);
    sGroup->setBuffer();
    groups.push_back(sGroup);
}
```

Figure 4: Parse de um grupo

Por último, renderizamos os pontos a partir da seguinte função:

```
void Group::render()
{
    glPushMatrix();

    for (int i = 0; i < transf.size(); i++)
        transf[i]->transform(primary);

    glBindBuffer(GL_ARRAY_BUFFER, buffer);
    glVertexPointer( size: 3, GL_FLOAT,  stride: 0,  pointer: 0);
    glDrawArrays(GL_TRIANGLES,  first: 0,  count: buffer_size * 3);

    for (int i = 0; i < groups.size(); i++)
        groups[i]->render();

    glPopMatrix();
}
```

Figure 5: Render de um grupo

2.2.2 Rotação dinâmica

De modo a permitir à nossa aplicação realizar rotações dinámicas, tivemos de realizar algumas alterações ao nosso código.

Primeiramente, chamamos às rotações anteriormente definidas de StaticRotation e criamos uma nova classe chamada DynamicRotation. Esta nova classe, em vez do angulo de rotação, terá como variável de instância o tempo, em milissegundos, que demora uma rotação completa a ser concluida.

Assim sendo, de modo a calcular o angulo de uma rotação dinámica, obtivemos o resultado da divisão inteira do tempo atual do programa, calculado pela função *glutGet(GLUT_ELAPSED_TIME)*, pelo tempo que demora uma rotação a ser concluída. Este valor é depois dividido pelo tempo que uma volta demora a ser realizada e multiplicado por 360 para obter um valor em graus.

```
// StaticRotation
StaticRotation::StaticRotation(float a, float xx, float yy, float zz) : Transformation(xx, yy, zz) {
    angle = a;
}

void StaticRotation::transform(bool primary){
    glRotatef(angle,  x: getX(),  y: getY(),  z: getZ());
}
```

Figure 6: StaticRotation

```cpp
// DynamicRotation
DynamicRotation::DynamicRotation(float t, float xx, float yy, float zz) : Transformation(xx, yy, zz) {
    time = t;
}

void DynamicRotation::transform(bool primary){
    float angle = (glutGet(GLUT_ELAPSED_TIME) % ((int) time)) / time * 360;

    glRotatef(angle,  x: getX(),  y: getY(),  z: getZ());
}
```

Figure 7: DynamicRotation

De modo a verificar que tipo de rotação é pretendida, no parse verificamos se o tempo para concluir uma rotação é 0. Caso seja verdade, trata-se de uma rotação estática, casa contrário, trata-se de uma rotação dinâmica.

```cpp
Transformation *parseRotate(XMLElement *element)
{
    float x = (element->Attribute( name: "axisX") ? stof( str: element->Attribute( name: "axisX")) : 0);
    float y = (element->Attribute( name: "axisY") ? stof( str: element->Attribute( name: "axisY")) : 0);
    float z = (element->Attribute( name: "axisZ") ? stof( str: element->Attribute( name: "axisZ")) : 0);
    float time = (element->Attribute( name: "time") ? stof( str: element->Attribute( name: "time")) : 0);

    if(time==0) {
        float angle = (element->Attribute( name: "angle") ? stof( str: element->Attribute( name: "angle")) : 0);
        return new StaticRotation(angle, x, y, z);
    } else
        return new DynamicRotation( t: time*1000, x, y, z);
}
```

Figure 8: Parse de uma rotação

2.2.3 Translação dinâmica

Para implemetar translações dinámicas, seguimos a mesma lógica que no tópico anterior, ou seja, chamamos à antiga classe *Translation* de *StaticTranslation* e criamos uma nova chamada *DynamicTranslation*.

```cpp
class DynamicTranslation : public Transformation {
    float time;
    vector<Point*> catmull;
    vector<Point*> orbit;
public:
    DynamicTranslation(float t, vector<Point*> catmullPoints);
    void transform(bool primary);
    vector<Point*> getCatmullPoints();

    void catmullRomPoint(float t, float *coord, vector<Point *> catmullpoints);

    void catmullCalculate(float t, int *index, float *p, vector<Point *> points);

    void drawCurve();
};

class StaticTranslation : public Transformation {
public:
    StaticTranslation(float xx, float yy, float zz);
    void transform(bool primary);
};
```

Figure 9: Classes *StaticTranslation* e *DynamicTranslation*

Para conseguirmos gerar os pontos por onde um objeto poderá realizar a sua translação utilizamos as funções de catmull. Para isso, no nosso ficheiro XML enviamos 16 pontos possíveis da orbita do planeta e através das funções de catmull conseguimos gerar uma aproximação à orbita desse planeta.

```
void DynamicTranslation::catmullRomPoint(float t, float* coord, vector<Point*> catmullpoints){
    int size = catmullpoints.size();

    float rt = t*size;
    int i = floor(rt);
    rt = rt - i;

    int index[4];
    index[0] = (i  + size - 1) % size;
    index[1] = (index[0]  + 1) % size;
    index[2] = (index[1]  + 1) % size;
    index[3] = (index[2]  + 1) % size;

    catmullCalculate(rt,index,coord,catmullpoints);
}
```

Figure 10: Função de catmull

```
void DynamicTranslation::catmullCalculate(float t, int* index, float* p, std::vector<Point*> points){
    float aux[4];
    float t_2 = t*t;
    float t_3 = t*t*t;

    float m[4][4] = { { -0.5f, 1.5f, -1.5f, 0.5f },
                      { 1.0f, -2.5f, 2.0f, -0.5f },
                      { -0.5f, 0.0f, 0.5f, 0.0f },
                      { 0.0f, 1.0f, 0.0f, 0.0f }};

    p[0] = 0.0;
    p[1] = 0.0;
    p[2] = 0.0;

    aux[0] = t_3*m[0][0] + t_2*m[1][0] + t*m[2][0] + m[3][0];
    aux[1] = t_3*m[0][1] + t_2*m[1][1] + t*m[2][1] + m[3][1];
    aux[2] = t_3*m[0][2] + t_2*m[1][2] + t*m[2][2] + m[3][2];
    aux[3] = t_3*m[0][3] + t_2*m[1][3] + t*m[2][3] + m[3][3];


    int i_0 = index[0];
    int i_1 = index[1];
    int i_2 = index[2];
    int i_3 = index[3];
    Point* v0 = points[i_0];
    Point* v1 = points[i_1];
    Point* v2 = points[i_2];
    Point* v3 = points[i_3];

    p[0] = aux[0] * v0->getX() + aux[1] * v1->getX() + aux[2] * v2->getX() + aux[3] * v3->getX();
    p[1] = aux[0] * v0->getY() + aux[1] * v1->getY() + aux[2] * v2->getY() + aux[3] * v3->getY();
    p[2] = aux[0] * v0->getZ() + aux[1] * v1->getZ() + aux[2] * v2->getZ() + aux[3] * v3->getZ();
}
```

Figure 11: Função de catmull

Para verificar que tipo de translação é pretendida, no parse verificamos se o tempo para concluir uma translação é 0. Caso seja verdade, trata-se de uma translação estática, casa contrário, trata-se de uma translação dinâmica.

```
Transformation *parseTranslate(XMLElement *element)
{
    float time = (element->Attribute( name: "time") ? stof( str: element->Attribute( name: "time")) : 0);

    if(time==0) {
        float x = (element->Attribute( name: "X") ? stof( str: element->Attribute( name: "X")) : 0);
        float y = (element->Attribute( name: "Y") ? stof( str: element->Attribute( name: "Y")) : 0);
        float z = (element->Attribute( name: "Z") ? stof( str: element->Attribute( name: "Z")) : 0);

        return new StaticTranslation(x, y, z);
    } else {
        vector<Point*> catmull;
        for (XMLElement *elem = element->FirstChildElement( name: "point"); elem; elem = elem->NextSiblingElement( name: "point")) {
            float x = (elem->Attribute( name: "X") ? stof( str: elem->Attribute( name: "X")) : 0);
            float y = (elem->Attribute( name: "Y") ? stof( str: elem->Attribute( name: "Y")) : 0);
            float z = (elem->Attribute( name: "Z") ? stof( str: elem->Attribute( name: "Z")) : 0);

            Point* p = new Point(x, y, z);
            catmull.push_back(p);
        }
        return new DynamicTranslation( t: time*1000, catmull);
    }
}
```

Figure 12: Parse de uma translação

2.2.4 Órbitas

De modo a gerar as órbitas dos nossos planetas, no construtor da *DynamicTranslation* geramos vários pontos através das funções de catmull explicadas anteriormente e guardamos num vetor *orbit*.

```
float p[3];

for (float gt = 0; gt < 1; gt += 0.01) {
    catmullRomPoint(gt, p, catmullpoints: getCatmullPoints());
    orbit.push_back(new Point( x1: p[0], y1: p[1], z1: p[2]));
}
```

Figure 13: Gerar as órbitas

Posteriormente, quando aplicamos a transformação referida, desenhamos as órbitas do planeta, caso esta seja um planeta primário.

```
if(primary) {
    glPushMatrix();
    glColor3f( red: 1, green: 1, blue: 1);
    drawCurve();
    glPopMatrix();
}
```

Figure 14: Desenhar as órbitas

```
void DynamicTranslation::drawCurve() {
    glBegin(GL_LINE_LOOP);

    for (int i = 0; i<orbit.size(); i++)
        glVertex3f( x: orbit[i]->getX(), y: orbit[i]->getY(), z: orbit[i]->getZ());

    glEnd();
}
```

Figure 15: Desenhar as órbitas

3 Ficheiro XML

Nesta fase, o ficheiro *XML* foi alvo de bastantes alterações, de modo a tornar o nosso Sistema Solar dinâmico. O esquema hierárquico do nosso sistema mantém-se, em que o astro central é o Sol e o restante sistema é criado em função deste. As escalas foram alteradas de modo a obter uma melhor visualização do conjunto em si e, também, para permitir uma rendirização mais facilitada e garantir que não ocorreriam falhas nesse mesmo processo. Foram, ainda, implementados os movimentos de translação e de rotação dos diversos astros, tendo sempre em conta a noção do tempo alusivo a cada um deles, que varia de componente para componente consoante os períodos relativos.

Além disso, também adicionamos um grupo relativo ao novo cometa gerado com o teapot.

Abaixo encontra-se um extração deste mesmo ficheiro *XML*, utilizando o planeta terra como exemplo:

```
<translate time="30.0">
    <point X="150.0" Z="0"/>
    <point X="138.58192987669298" Z="-57.40251485476346"/>
    <point X="106.06601717798213" Z="-106.06601717798213"/>
    <point X="57.40251485476346" Z="-138.58192987669298"/>
    <point X="0" Z="-150.0"/>
    <point X="-57.40251485476346" Z="-138.58192987669298"/>
    <point X="-106.06601717798213" Z="-106.06601717798213"/>
    <point X="-138.58192987669298" Z="-57.40251485476346"/>
    <point X="-150.0" Z="0"/>
    <point X="-138.58192987669298" Z="57.40251485476346"/>
    <point X="-106.06601717798213" Z="106.06601717798213"/>
    <point X="-57.40251485476346" Z="138.58192987669298"/>
    <point X="0" Z="150.0"/>
    <point X="57.40251485476346" Z="138.58192987669298"/>
    <point X="106.06601717798213" Z="106.06601717798213"/>
    <point X="138.58192987669298" Z="57.40251485476346"/>
</translate>
<rotate angle="23.5" axisX="1" />
<rotate time="10.0" axisY="1" />
<scale X="2.6774999999999998" Y="2.6774999999999998" Z="2.6774999999999998" />
<colour R="61" G="129" B="224" />
<models>
    <model file="sphere.3d"/>
</models>
```

Figure 16: XML

4 Demonstração

4.1 Generator

Para a executar a demonstração do Sistema Solar é necessário gerar 4 figuras: sphere.3d, ring.3d, uranusRing.3d e teapot.3d. Nas figuras em baixo indicamos como gerar essas figuras.

```
~/G/U/3/2/CG/CG_Project/cmake-build-debug main > ./generator sphere 1 100 100 sphere.3d
File: sphere.3d
```

Figure 17: Comando para gerar o ficheiro *sphere.3d*

```
~/G/U/3/2/CG/CG_Project/cmake-build-debug main > ./generator torus 0.1 1 100 100 ring.3d
File: ring.3d
```

Figure 18: Comando para gerar o ficheiro *ring.3d*

```
~/G/U/3/2/CG/CG_Project/cmake-build-debug main > ./generator torus 0.0005 1 100 100 uranusRing.3d
File: uranusRing.3d
```

Figure 19: Comando para gerar o ficheiro *uranusRing.3d*

```
~/G/U/3/2/CG/CG_Project/cmake-build-debug main > ./generator bezier ../teapot.patch 2 teapot.3d
File: teapot.3d
```

Figure 20: Comando para gerar o ficheiro *teapot.3d*

4.2 Engine

Como é possível ver na figura abaixo, para representar o sistema solar temos de passar como parâmetro o ficheiro test.xml ao correr o engine. Assim, o sistema irá renderizar o sistema de acordo com os ficheiros .3d gerados anteriormente.

```
~/G/U/3/2/CG/CG_Project/cmake-build-debug main > ./engine test.xml
```

Figure 21: Comando para correr o engine

4.3 Sistema Solar

Nesta versão do projeto, de forma a reduzir a complexidade do sistema solar, decidimos remover os cinturões de asteroides e de kuiper, bem como remover algumas luas e reduzir a escala do sistema em geral.

Por outro lado, adicionamos, além de movimentos de rotação e translação, um cometa representado por um tpot e órbitas geradas pelas funções de catmull.

De seguida, segue-se uma demonstração do nosso sistema solar:

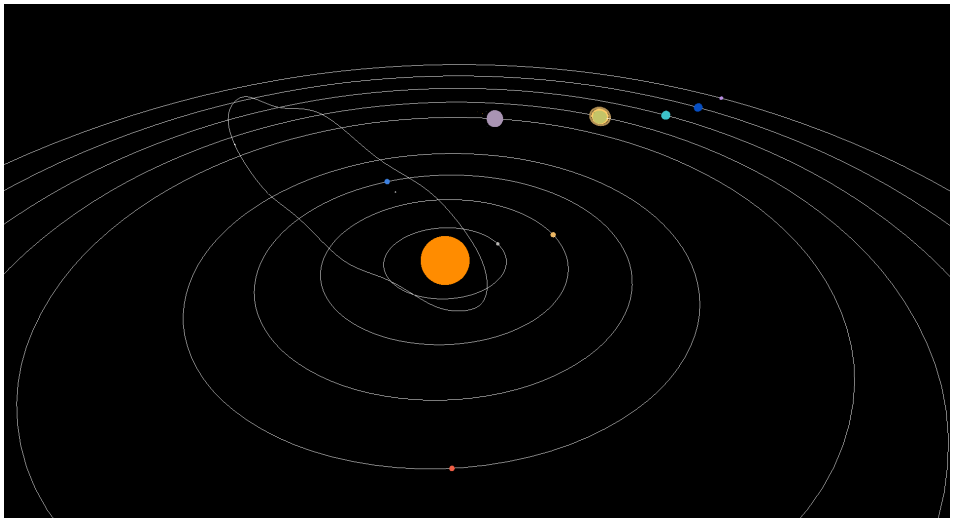

Figure 22: Sistema Solar completo

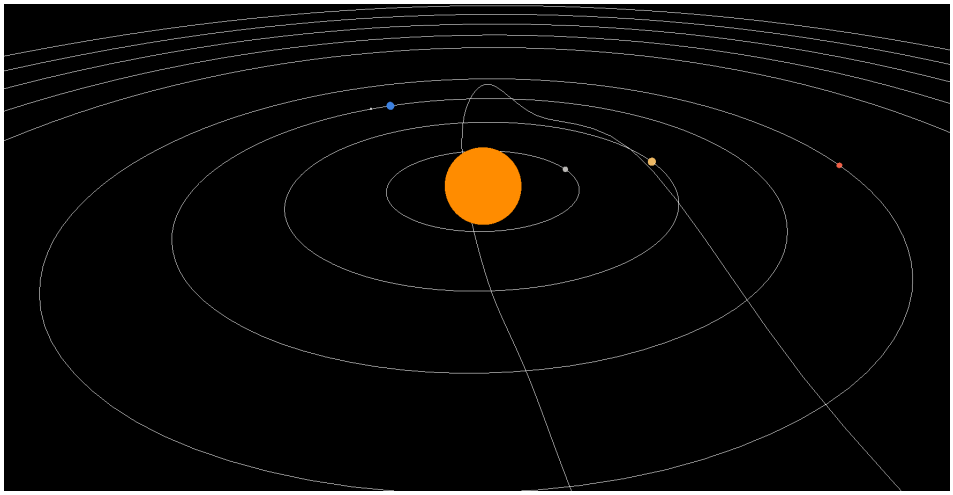

Figure 23: Planetas Sólidos e Sol

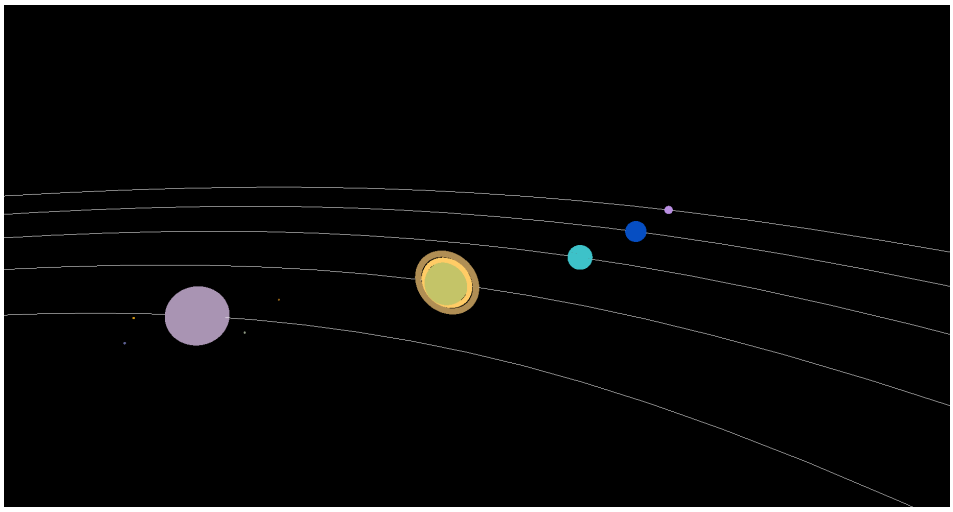

Figure 24: Planetas Gasosos

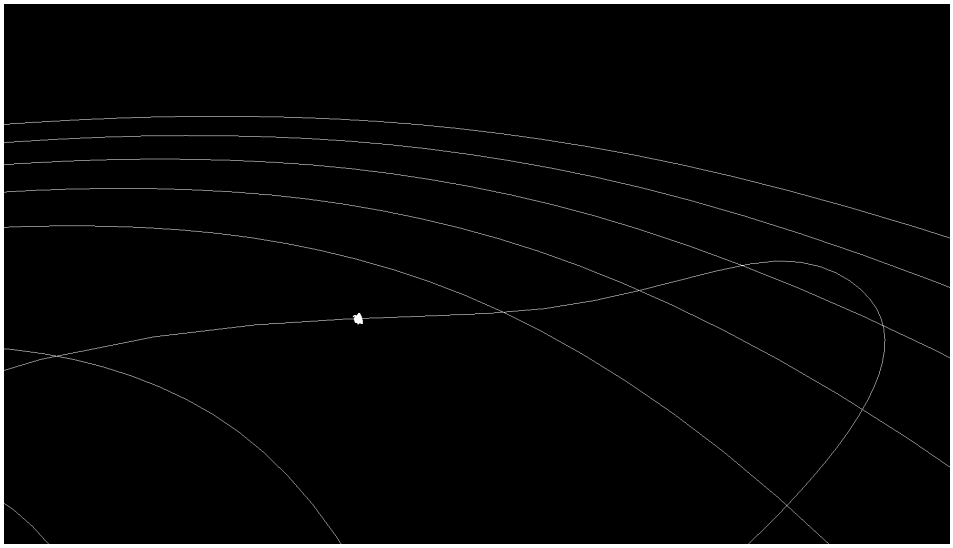

Figure 25: Cometa

5 Conclusão e Trabalho Futuro

Nesta segunda fase, foi novamente necessária uma reformulação de algum do trabalho feito anteriormente a fim de suportar melhor tudo aquilo que pretendíamos implementar relativamente ao Sistema Solar.

Para a construção do Sistema Solar, o ficheiro XML sofreu bastantes alterações e criamos uma nova figura de um cometa que implementamos no Sistema Solar.

De uma perspetiva geral, consideramos que esta fase foi bem sucedida, uma vez que acreditamos ter cumprido com todos os requisitos estabelecidos inicialmente.

Assim, consideramos que foi alcançada uma melhor consolidação da matéria lecionada nas aulas, sobre a utilização e domínio de ferramentas e conceitos associados à Computação Gráfica, permitindo, também, a exploração de domínios totalmente novos.

A Anexos

A.1 Ficheiro XML - Sistema Solar

```
<scene>
    <!-- Sol-->
    <group>
        <rotate angle="7" axisX="1" />
        <rotate time="250.0" axisY="1" />
        <scale X="20.0" Y="20.0" Z="20.0" />
        <colour R="255" G="140" B="0" />
        <models>
            <model file="sphere.3d"/>
        </models>
    </group>
    <!-- Mercurio-->
    <group>
        <translate time="7.232876712328767">
            <point X="50.0" Z="0"/>
            <point X="46.19397662556433" Z="-19.134171618254488"/>
            <point X="35.35533905932738" Z="-35.35533905932738"/>
            <point X="19.134171618254488" Z="-46.19397662556433"/>
            <point X="0" Z="-50.0"/>
            <point X="-19.134171618254488" Z="-46.19397662556433"/>
            <point X="-35.35533905932738" Z="-35.35533905932738"/>
            <point X="-46.19397662556433" Z="-19.134171618254488"/>
            <point X="-50.0" Z="0"/>
            <point X="-46.19397662556433" Z="19.134171618254488"/>
            <point X="-35.35533905932738" Z="35.35533905932738"/>
            <point X="-19.134171618254488" Z="46.19397662556433"/>
            <point X="0" Z="50.0"/>
            <point X="19.134171618254488" Z="46.19397662556433"/>
            <point X="35.35533905932738" Z="35.35533905932738"/>
            <point X="46.19397662556433" Z="19.134171618254488"/>
        </translate>
        <rotate time="586.6666666666666" axisY="1" />
        <scale X="1.5" Y="1.5" Z="1.5" />
        <colour R="186" G="184" B="181" />
        <models>
            <model file="sphere.3d"/>
        </models>
    </group>
    <!-- Venus-->
    <group>
        <translate time="18.493150684931507">
            <point X="100.0" Z="0"/>
            <point X="92.38795325112866" Z="-38.268343236508976"/>
            <point X="70.71067811865476" Z="-70.71067811865476"/>
            <point X="38.268343236508976" Z="-92.38795325112866"/>
            <point X="0" Z="-100.0"/>
            <point X="-38.268343236508976" Z="-92.38795325112866"/>
            <point X="-70.71067811865476" Z="-70.71067811865476"/>
```

```
            <point X="-92.38795325112866" Z="-38.268343236508976"/>
            <point X="-100.0" Z="0"/>
            <point X="-92.38795325112866" Z="38.268343236508976"/>
            <point X="-70.71067811865476" Z="70.71067811865476"/>
            <point X="-38.268343236508976" Z="92.38795325112866"/>
            <point X="0" Z="100.0"/>
            <point X="38.268343236508976" Z="92.38795325112866"/>
            <point X="70.71067811865476" Z="70.71067811865476"/>
            <point X="92.38795325112866" Z="38.268343236508976"/>
        </translate>
        <rotate angle="-3" axisX="1" />
        <rotate time="2430.0" axisY="1" />
        <scale X="2.25" Y="2.25" Z="2.25" />
        <colour R="237" G="184" B="100" />
        <models>
            <model file="sphere.3d"/>
        </models>
    </group>
    <!-- Terra-->
    <group>
        <translate time="30.0">
            <point X="150.0" Z="0"/>
            <point X="138.58192987669298" Z="-57.40251485476346"/>
            <point X="106.06601717798213" Z="-106.06601717798213"/>
            <point X="57.40251485476346" Z="-138.58192987669298"/>
            <point X="0" Z="-150.0"/>
            <point X="-57.40251485476346" Z="-138.58192987669298"/>
            <point X="-106.06601717798213" Z="-106.06601717798213"/>
            <point X="-138.58192987669298" Z="-57.40251485476346"/>
            <point X="-150.0" Z="0"/>
            <point X="-138.58192987669298" Z="57.40251485476346"/>
            <point X="-106.06601717798213" Z="106.06601717798213"/>
            <point X="-57.40251485476346" Z="138.58192987669298"/>
            <point X="0" Z="150.0"/>
            <point X="57.40251485476346" Z="138.58192987669298"/>
            <point X="106.06601717798213" Z="106.06601717798213"/>
            <point X="138.58192987669298" Z="57.40251485476346"/>
        </translate>
        <rotate angle="23.5" axisX="1" />
        <rotate time="10.0" axisY="1" />
        <scale X="2.6774999999999998" Y="2.6774999999999998" Z="2.6774999999999998" />
        <colour R="61" G="129" B="224" />
        <models>
            <model file="sphere.3d"/>
        </models>
        <!-- Lua-->
        <group>
            <translate time="2.219178082191781">
                <point X="5.602240896358544" Z="0"/>
                <point X="5.175795700343343" Z="-2.1438847751545644"/>
                <point X="3.9613825276557293" Z="-3.9613825276557293"/>
```

```
                <point X="2.1438847751545644" Z="-5.175795700343343"/>
                <point X="0" Z="-5.602240896358544"/>
                <point X="-2.1438847751545644" Z="-5.175795700343343"/>
                <point X="-3.9613825276557293" Z="-3.9613825276557293"/>
                <point X="-5.175795700343343" Z="-2.1438847751545644"/>
                <point X="-5.602240896358544" Z="0"/>
                <point X="-5.175795700343343" Z="2.1438847751545644"/>
                <point X="-3.9613825276557293" Z="3.9613825276557293"/>
                <point X="-2.1438847751545644" Z="5.175795700343343"/>
                <point X="0" Z="5.602240896358544"/>
                <point X="2.1438847751545644" Z="5.175795700343343"/>
                <point X="3.9613825276557293" Z="3.9613825276557293"/>
                <point X="5.175795700343343" Z="2.1438847751545644"/>
            </translate>
            <rotate angle="6.68" axisX="1" />
            <rotate time="12.291666666666666" axisY="1" />
            <scale X="0.2801120448179272" Y="0.2801120448179272" Z="0.2801120448179272" />
            <colour R="207" G="207" B="207" />
            <models>
                <model file="sphere.3d"/>
            </models>
        </group>
    </group>
    <!-- Marte-->
    <group>
        <translate time="56.465753424657535">
            <point X="200.0" Z="0"/>
            <point X="184.77590650225733" Z="-76.53668647301795"/>
            <point X="141.4213562373095" Z="-141.4213562373095"/>
            <point X="76.53668647301795" Z="-184.77590650225733"/>
            <point X="0" Z="-200.0"/>
            <point X="-76.53668647301795" Z="-184.77590650225733"/>
            <point X="-141.4213562373095" Z="-141.4213562373095"/>
            <point X="-184.77590650225733" Z="-76.53668647301795"/>
            <point X="-200.0" Z="0"/>
            <point X="-184.77590650225733" Z="76.53668647301795"/>
            <point X="-141.4213562373095" Z="141.4213562373095"/>
            <point X="-76.53668647301795" Z="184.77590650225733"/>
            <point X="0" Z="200.0"/>
            <point X="76.53668647301795" Z="184.77590650225733"/>
            <point X="141.4213562373095" Z="141.4213562373095"/>
            <point X="184.77590650225733" Z="76.53668647301795"/>
        </translate>
        <rotate angle="25" axisX="1" />
        <rotate time="10.416666666666666" axisY="1" />
        <scale X="1.5" Y="1.5" Z="1.5" />
        <colour R="240" G="95" B="70" />
        <models>
            <model file="sphere.3d"/>
        </models>
    </group>
```

```
<!-- Jupiter-->
<group>
    <translate time="360.0">
        <point X="300.0" Z="0"/>
        <point X="277.16385975338596" Z="-114.80502970952692"/>
        <point X="212.13203435596427" Z="-212.13203435596427"/>
        <point X="114.80502970952692" Z="-277.16385975338596"/>
        <point X="0" Z="-300.0"/>
        <point X="-114.80502970952692" Z="-277.16385975338596"/>
        <point X="-212.13203435596427" Z="-212.13203435596427"/>
        <point X="-277.16385975338596" Z="-114.80502970952692"/>
        <point X="-300.0" Z="0"/>
        <point X="-277.16385975338596" Z="114.80502970952692"/>
        <point X="-212.13203435596427" Z="212.13203435596427"/>
        <point X="-114.80502970952692" Z="277.16385975338596"/>
        <point X="0" Z="300.0"/>
        <point X="114.80502970952692" Z="277.16385975338596"/>
        <point X="212.13203435596427" Z="212.13203435596427"/>
        <point X="277.16385975338596" Z="114.80502970952692"/>
    </translate>
    <rotate angle="3" axisX="1" />
    <rotate time="4.166666666666667" axisY="1" />
    <scale X="10.0" Y="10.0" Z="10.0" />
    <colour R="169" G="148" B="179" />
    <models>
        <model file="sphere.3d"/>
    </models>
    <!-- Io-->
    <group>
        <translate time="0.14383561643835616">
            <point X="2.0" Z="0"/>
            <point X="1.8477590650225733" Z="-0.7653668647301795"/>
            <point X="1.4142135623730951" Z="-1.4142135623730951"/>
            <point X="0.7653668647301795" Z="-1.8477590650225733"/>
            <point X="0" Z="-2.0"/>
            <point X="-0.7653668647301795" Z="-1.8477590650225733"/>
            <point X="-1.4142135623730951" Z="-1.4142135623730951"/>
            <point X="-1.8477590650225733" Z="-0.7653668647301795"/>
            <point X="-2.0" Z="0"/>
            <point X="-1.8477590650225733" Z="0.7653668647301795"/>
            <point X="-1.4142135623730951" Z="1.4142135623730951"/>
            <point X="-0.7653668647301795" Z="1.8477590650225733"/>
            <point X="0" Z="2.0"/>
            <point X="0.7653668647301795" Z="1.8477590650225733"/>
            <point X="1.4142135623730951" Z="1.4142135623730951"/>
            <point X="1.8477590650225733" Z="0.7653668647301795"/>
        </translate>
        <rotate time="17.7" axisY="1" />
        <scale X="0.0375" Y="0.0375" Z="0.0375" />
        <colour R="209" G="150" B="23" />
        <models>
```

```
            <model file="sphere.3d"/>
        </models>
    </group>
    <!-- Europa-->
    <group>
        <translate time="0.290958904109589">
            <point X="2.25" Z="0"/>
            <point X="2.078728948150395" Z="-0.8610377228214519"/>
            <point X="1.5909902576697321" Z="-1.5909902576697321"/>
            <point X="0.8610377228214519" Z="-2.078728948150395"/>
            <point X="0" Z="-2.25"/>
            <point X="-0.8610377228214519" Z="-2.078728948150395"/>
            <point X="-1.5909902576697321" Z="-1.5909902576697321"/>
            <point X="-2.078728948150395" Z="-0.8610377228214519"/>
            <point X="-2.25" Z="0"/>
            <point X="-2.078728948150395" Z="0.8610377228214519"/>
            <point X="-1.5909902576697321" Z="1.5909902576697321"/>
            <point X="-0.8610377228214519" Z="2.078728948150395"/>
            <point X="0" Z="2.25"/>
            <point X="0.8610377228214519" Z="2.078728948150395"/>
            <point X="1.5909902576697321" Z="1.5909902576697321"/>
            <point X="2.078728948150395" Z="0.8610377228214519"/>
        </translate>
        <rotate time="35.0" axisY="1" />
        <scale X="0.0375" Y="0.0375" Z="0.0375" />
        <colour R="141" G="152" B="131" />
        <models>
            <model file="sphere.3d"/>
        </models>
    </group>
    <!-- Ganymede-->
    <group>
        <translate time="0.5884931506849316">
            <point X="2.5" Z="0"/>
            <point X="2.3096988312782165" Z="-0.9567085809127243"/>
            <point X="1.7677669529663689" Z="-1.7677669529663689"/>
            <point X="0.9567085809127243" Z="-2.3096988312782165"/>
            <point X="0" Z="-2.5"/>
            <point X="-0.9567085809127243" Z="-2.3096988312782165"/>
            <point X="-1.7677669529663689" Z="-1.7677669529663689"/>
            <point X="-2.3096988312782165" Z="-0.9567085809127243"/>
            <point X="-2.5" Z="0"/>
            <point X="-2.3096988312782165" Z="0.9567085809127243"/>
            <point X="-1.7677669529663689" Z="1.7677669529663689"/>
            <point X="-0.9567085809127243" Z="2.3096988312782165"/>
            <point X="0" Z="2.5"/>
            <point X="0.9567085809127243" Z="2.3096988312782165"/>
            <point X="1.7677669529663689" Z="1.7677669529663689"/>
            <point X="2.3096988312782165" Z="0.9567085809127243"/>
        </translate>
        <rotate time="70.0" axisY="1" />
```

```
            <scale X="0.0375" Y="0.0375" Z="0.0375" />
            <colour R="95" G="99" B="150" />
            <models>
                <model file="sphere.3d"/>
            </models>
        </group>
        <!-- Calisto-->
        <group>
            <translate time="1.3972602739726028">
                <point X="2.75" Z="0"/>
                <point X="2.540668714406038" Z="-1.0523794390039967"/>
                <point X="1.9445436482630059" Z="-1.9445436482630059"/>
                <point X="1.0523794390039967" Z="-2.540668714406038"/>
                <point X="0" Z="-2.75"/>
                <point X="-1.0523794390039967" Z="-2.540668714406038"/>
                <point X="-1.9445436482630059" Z="-1.9445436482630059"/>
                <point X="-2.540668714406038" Z="-1.0523794390039967"/>
                <point X="-2.75" Z="0"/>
                <point X="-2.540668714406038" Z="1.0523794390039967"/>
                <point X="-1.9445436482630059" Z="1.9445436482630059"/>
                <point X="-1.0523794390039967" Z="2.540668714406038"/>
                <point X="0" Z="2.75"/>
                <point X="1.0523794390039967" Z="2.540668714406038"/>
                <point X="1.9445436482630059" Z="1.9445436482630059"/>
                <point X="2.540668714406038" Z="1.0523794390039967"/>
            </translate>
            <rotate time="160.0" axisY="1" />
            <scale X="0.0375" Y="0.0375" Z="0.0375" />
            <colour R="150" G="101" B="24" />
            <models>
                <model file="sphere.3d"/>
            </models>
        </group>
    </group>
    <!-- Saturno-->
    <group>
        <translate time="870.0">
            <point X="350.0" Z="0"/>
            <point X="323.3578363789503" Z="-133.9392013277814"/>
            <point X="247.48737341529164" Z="-247.48737341529164"/>
            <point X="133.9392013277814" Z="-323.3578363789503"/>
            <point X="0" Z="-350.0"/>
            <point X="-133.9392013277814" Z="-323.3578363789503"/>
            <point X="-247.48737341529164" Z="-247.48737341529164"/>
            <point X="-323.3578363789503" Z="-133.9392013277814"/>
            <point X="-350.0" Z="0"/>
            <point X="-323.3578363789503" Z="133.9392013277814"/>
            <point X="-247.48737341529164" Z="247.48737341529164"/>
            <point X="-133.9392013277814" Z="323.3578363789503"/>
            <point X="0" Z="350.0"/>
            <point X="133.9392013277814" Z="323.3578363789503"/>
```

```
            <point X="247.48737341529164" Z="247.48737341529164"/>
            <point X="323.3578363789503" Z="133.9392013277814"/>
        </translate>
        <rotate angle="27" axisX="1" />
        <rotate time="4.583333333333333" axisY="1" />
        <scale X="8.5" Y="8.5" Z="8.5" />
        <colour R="196" G="196" B="104" />
        <models>
            <model file="sphere.3d"/>
        </models>
        <!-- Inner_rings-->
        <group>
            <scale X="1.125" Y="1.125" Z="1.125" />
            <colour R="255" G="204" B="102" />
            <models>
                <model file="ring.3d"/>
            </models>
        </group>
        <!-- Outer_rings-->
        <group>
            <scale X="1.4249999999999998" Y="1.4249999999999998" Z="1.4249999999999998" />
            <colour R="176" G="142" B="84" />
            <models>
                <model file="ring.3d"/>
            </models>
        </group>
    </group>
    <!-- Urano-->
    <group>
        <translate time="2520.0">
            <point X="400.0" Z="0"/>
            <point X="369.55181300451466" Z="-153.0733729460359"/>
            <point X="282.842712474619" Z="-282.842712474619"/>
            <point X="153.0733729460359" Z="-369.55181300451466"/>
            <point X="0" Z="-400.0"/>
            <point X="-153.0733729460359" Z="-369.55181300451466"/>
            <point X="-282.842712474619" Z="-282.842712474619"/>
            <point X="-369.55181300451466" Z="-153.0733729460359"/>
            <point X="-400.0" Z="0"/>
            <point X="-369.55181300451466" Z="153.0733729460359"/>
            <point X="-282.842712474619" Z="282.842712474619"/>
            <point X="-153.0733729460359" Z="369.55181300451466"/>
            <point X="0" Z="400.0"/>
            <point X="153.0733729460359" Z="369.55181300451466"/>
            <point X="282.842712474619" Z="282.842712474619"/>
            <point X="369.55181300451466" Z="153.0733729460359"/>
        </translate>
        <rotate angle="-82" axisX="1" />
        <rotate time="7.083333333333333" axisY="1" />
        <scale X="5.324999999999999" Y="5.324999999999999" Z="5.324999999999999" />
        <colour R="61" G="194" B="201" />
```

```
        <models>
            <model file="sphere.3d"/>
        </models>
        <!-- Rings-->
        <group>
            <scale X="1.1267605633802817" Y="1.1267605633802817" Z="1.1267605633802817" />
            <colour R="100" G="100" B="100" />
            <models>
                <model file="uranusRing.3d"/>
            </models>
        </group>
    </group>
    <!-- Neptuno-->
    <group>
        <translate time="4950.0">
            <point X="450.0" Z="0"/>
            <point X="415.745789630079" Z="-172.20754456429037"/>
            <point X="318.1980515339464" Z="-318.1980515339464"/>
            <point X="172.20754456429037" Z="-415.745789630079"/>
            <point X="0" Z="-450.0"/>
            <point X="-172.20754456429037" Z="-415.745789630079"/>
            <point X="-318.1980515339464" Z="-318.1980515339464"/>
            <point X="-415.745789630079" Z="-172.20754456429037"/>
            <point X="-450.0" Z="0"/>
            <point X="-415.745789630079" Z="172.20754456429037"/>
            <point X="-318.1980515339464" Z="318.1980515339464"/>
            <point X="-172.20754456429037" Z="415.745789630079"/>
            <point X="0" Z="450.0"/>
            <point X="172.20754456429037" Z="415.745789630079"/>
            <point X="318.1980515339464" Z="318.1980515339464"/>
            <point X="415.745789630079" Z="172.20754456429037"/>
        </translate>
        <rotate angle="30" axisX="1" />
        <rotate time="6.666666666666667" axisY="1" />
        <scale X="5.1" Y="5.1" Z="5.1" />
        <colour R="6" G="78" B="194" />
        <models>
            <model file="sphere.3d"/>
        </models>
    </group>
    <!-- Plutao-->
    <group>
        <translate time="7440.0">
            <point X="500.0" Z="0"/>
            <point X="461.9397662556433" Z="-191.34171618254487"/>
            <point X="353.5533905932738" Z="-353.5533905932738"/>
            <point X="191.34171618254487" Z="-461.9397662556433"/>
            <point X="0" Z="-500.0"/>
            <point X="-191.34171618254487" Z="-461.9397662556433"/>
            <point X="-353.5533905932738" Z="-353.5533905932738"/>
            <point X="-461.9397662556433" Z="-191.34171618254487"/>
```

```
            <point X="-500.0" Z="0"/>
            <point X="-461.9397662556433" Z="191.34171618254487"/>
            <point X="-353.5533905932738" Z="353.5533905932738"/>
            <point X="-191.34171618254487" Z="461.9397662556433"/>
            <point X="0" Z="500.0"/>
            <point X="191.34171618254487" Z="461.9397662556433"/>
            <point X="353.5533905932738" Z="353.5533905932738"/>
            <point X="461.9397662556433" Z="191.34171618254487"/>
        </translate>
        <rotate angle="-72" axisX="1" />
        <rotate time="63.75" axisY="1" />
        <scale X="2.25" Y="2.25" Z="2.25" />
        <colour R="186" G="143" B="227" />
        <models>
            <model file="sphere.3d"/>
        </models>
    </group>
    <!-- Cometa -->
    <group>
        <translate time="50">
            <point Y="80" Z="-160" />
            <point Y="63.9" Z="-79.64" />
            <point Y="56.56" Z="-11.5" />
            <point Y="30.64" Z="34.02" />
            <point Y="0" Z="48" />
            <point Y="-30.64" Z="34.02" />
            <point Y="-56.56" Z="-11.5" />
            <point Y="-63.9" Z="-79.64" />
            <point Y="-80" Z="-160" />
            <point Y="-63.9" Z="-240.36" />
            <point Y="-56.56" Z="-308.5" />
            <point Y="-30.64" Z="-354.02" />
            <point Y="0" Z="-370" />
            <point Y="30.64" Z="-354.02" />
            <point Y="56.56" Z="-308.5" />
            <point Y="63.9" Z="-240.36" />
        </translate>
        <rotate angle="260" X="1" Y="0" Z="0"/>
        <scale X="0.5" Y="0.5" Z="0.5" />
        <models>
            <model file="teapot.3d"/>
        </models>
    </group>
</scene>
```