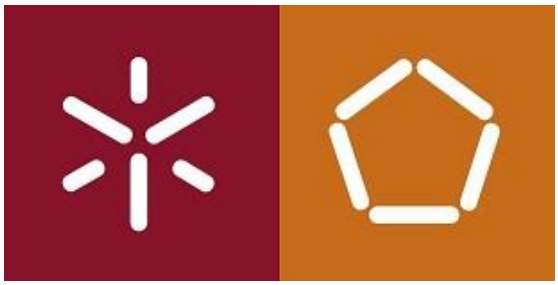

Universidade do Minho

Engenharia Informática

Trabalho Prático
Computação Gráfica

2º Semestre - 2021/2022

RELATÓRIO CG – Fase 1

Março 2022

Diogo da Costa e Silva Lima Rebelo (A93180)

Diogo Miguel da Silva Araújo (A93313)

Joel Costa Araújo (A76603)

Francisco Peixoto (A84668)

Resumo

O presente documento serve como guia para o trabalho efetuado na primeira fase deste projeto, no qual geramos os vértices de figuras propostas e os utilizamos para posterior geração de uma visualização dos modelos. O trabalho realizado focou-se na definição dos vértices apropriados para cada figura, assim como a sua disposição para posterior consumo, por parte de funcionalidades fornecidas pela biblioteca openGL.

Conteúdo

Índice de Ilustrações

Fase 1 – Primitivas Gráficas

Gerador

Para facilitar a definição de cada uma das primitivas que nos foi pedida, criamos, para cada figura, uma classe cujo trabalho individual seria calcular os pontos necessários à sua configuração. Nestes, além das coordenadas, fazemos acompanhá-los de uma lista de índices que servem para indicar quais as configurações de cada triângulo que constitui a figura.

O gerador serve como base para a geração de ficheiros .3d, que servem para a posterior configuração gráfica.

Nestes ficheiros, a primeira linha indica um código identificador da figura (plano – 01, box – 02, cone – 03, esfera – 04), acompanhada do número de pontos utilizados para a definir, assim como o número de indexes que auxiliam nessa configuração.

Plano

Para definir um plano seguimos uma metodologia que melhor se atenta ao visualizar a seguinte figura **(1)**.

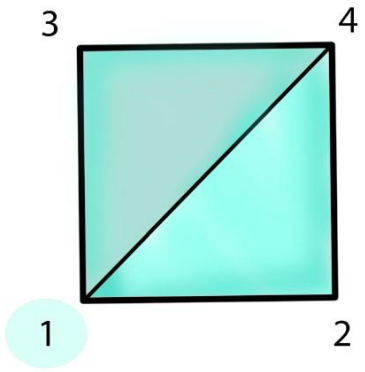

Figura 1 - Plano

A partir de uma representação simples, em 2D, identificamos um plano como um quadrado com 4 vértices limitadores e escolhemos o ponto representado em 1 como um ponto de referência, o qual utilizaríamos para definir o resto do plano.

Assumindo também as divisões requeridas por eixo (no caso da figura **(2)**, há apenas 1 divisão), para configurar graficamente o plano precisamos de unir os triângulos que o constituem, escolhendo como "front face" uma configuração CCW ("*counter clock wise*").

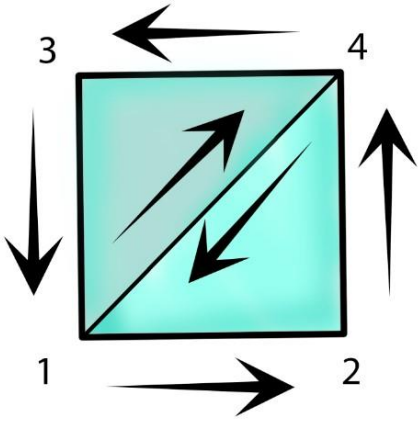

Figura 2 - Triângulos [1,4,3] e [1,2,4] (CCW)

Deste modo, e como representado na figura acima, para definir os triângulos que constituem o plano seguimos a ordem da esquerda para a direita, baixo para cima. Para representar o plano desta figura, configuramos os triângulos [1, 4, 3] e [1, 2, 4].

Como exemplo já definido por divisões, apresentamos também uma representação dos pontos que consideramos na sua construção, relembrando que constituímos os triângulos de baixo para cima, esquerda para a direita.

- Linha1: 1,6,5 e 1,2,6 (primeira utilizada)
- Linha3: 9,14,13 e 9,10,14 (última utilizada)

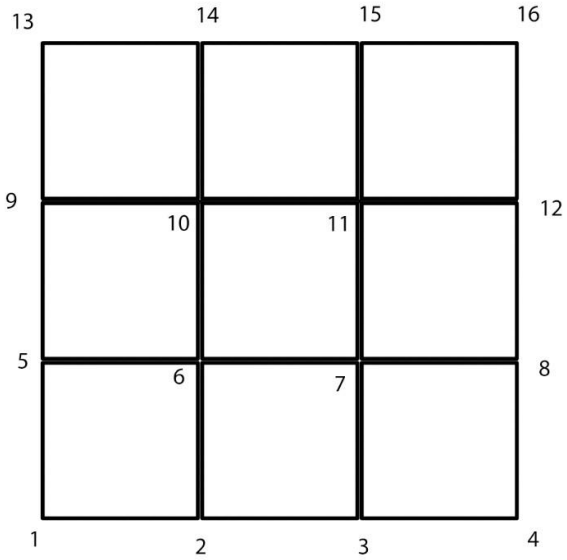

Figura 3 - Plano com divisões

Caixa

Para configurar a caixa foi utilizado, como base, o plano referido anteriormente. Visto que já se tratava das divisões e do tamanho no plano, nesta etapa foi apenas necessário transformar o plano "XZ" em planos "XY" e "YZ" através das funções "*planeXY*" e "*planeYZ*", respetivamente, gerando assim 3 faces da caixa figura**(4).**

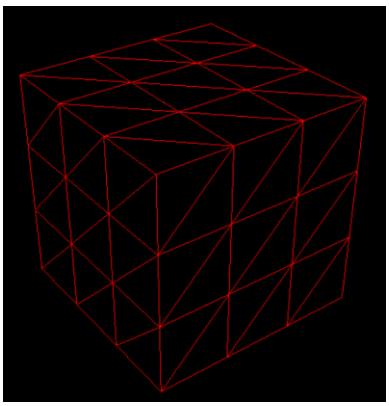

Figura 4 - Planos XZ, XY, YZ

Após obtermos as faces da frente do plano foi necessário inverter os planos para que estes ficassem com as mesmas viradas para a camara e pudessem ser vistas quando se roda o objeto. Para tal, criamos então a função *planeVerticesInverted* para utilizar os pontos já calculados nos planos anteriores com a pequena alteração de, em vez de desenhar os triângulos do plano [1,4,3] e [1,2,4], como na figura 2, desenha seguindo uma ordem inversa [1,4,2] e [1,3,4], como se pode ver na figura **(5)** abaixo. Deste modo, ficam obtidos os 6 planos necessários para representar a caixa.

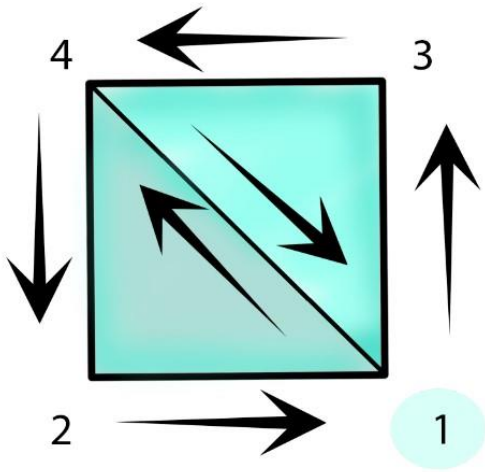

Figura 5 - Plano Invertido. Triângulos [1,4,2] e [1,3,4]

Cone

De forma a elaborar os pontos que permitem a constituição do cone, o grupo utilizou o processo das aulas práticas para obter os pontos de um cilindro, em que a base superior tem raio 0. O cone é formado através da medida do raio da base, o número de *slices* (que dividem igualmente as laterais do cone) e a medida da altura. O processo da formulação dos pontos do cone é dado através do cálculo das coordenadas polares, que depois são convertidas para coordenadas cartesianas.

Esfera

A nossa estratégia para encontrar os pontos de uma esfera de raio r foi, *stack* a *stack*, calcular os pontos das *slices* presentes nessa *stack*, visto que o valor do y é sempre o mesmo para todos os pontos de uma *stack*. Assim, resta calcular as coordenadas do x e do z através de conhecimentos de trigonometria.

O ângulo das *slices* pode variar de 0 a 360, enquanto o ângulo de uma *stack* varia entre -90 e +90 (onde +90 é o polo superior e -90 o inferior).

Após o cálculo dos pontos, o método de construção é semelhante ao das outras figuras, à exceção dos polos, que em vez de ser um quadrado composto por dois triângulos, é composto apenas por um, como se pode observar na figura **(6)**.

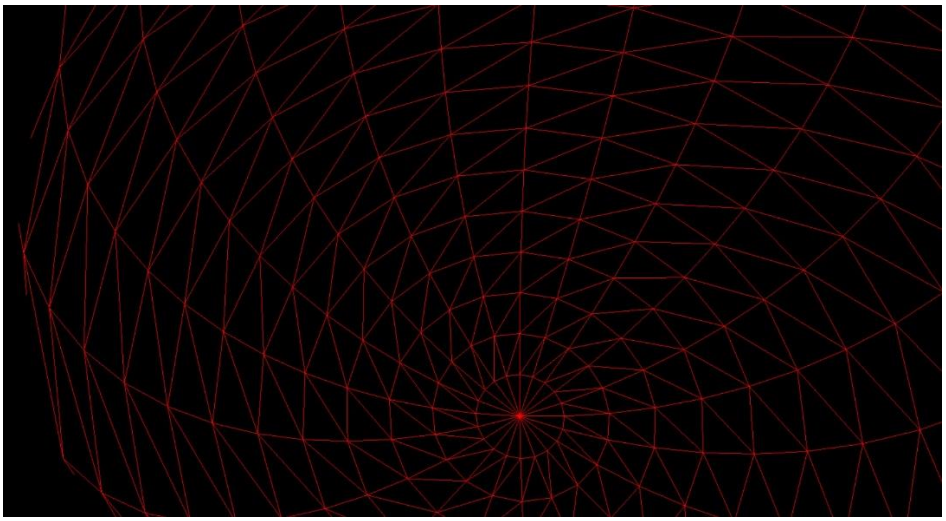

Figura 6 - Sphere Polos

Engine

A ferramenta *engine* é o motor gráfico que interpreta os ficheiros xml elaborados pelo grupo e recorrendo à estrutura dos ficheiros .3d desenha no ecrã as figuras pretendidas. O engine recorre ao uso de uma camara de forma a visualizar as figuras e navegar pelo espaço, sendo que as figuras são carregadas pelo programa uma única vez, e a cada *frame* é recalculado o que mostrar no ecrã. Para carregar as figuras elaborou-se um *parser* xml, com base nos exemplos dados, o *parser* devolve as informações a uma *class Scene* que preserva os pontos de cada figura bem como a cor respetiva, esta estrutura é modelar e futuramente podem-se incluir novas informações relevantes.

Conclusão

Dada por concluída esta primeira fase do projeto, foi possível compreender melhor o funcionamento de um motor gráfico 3D assim como diversas funcionalidades do OpenGL para gerar formas geométricas básicas e complexas. Conseguimos dar por concluído tudo o que foi proposto, com sucesso e satisfação, introduzindo ao projeto algumas funcionalidades extra como a escolha de cores nos objetos através do xml e a camara com modo terceira pessoa e primeira pessoa. O grupo entende que certos compromissos se tornaram evidentes no decorrer do trabalho, pelo que as estruturas finais do *generator* e do engine não são para já finais, mas o trabalho elaborado permite a modulação das componentes a alterar sem grande dificuldade.

Anexos

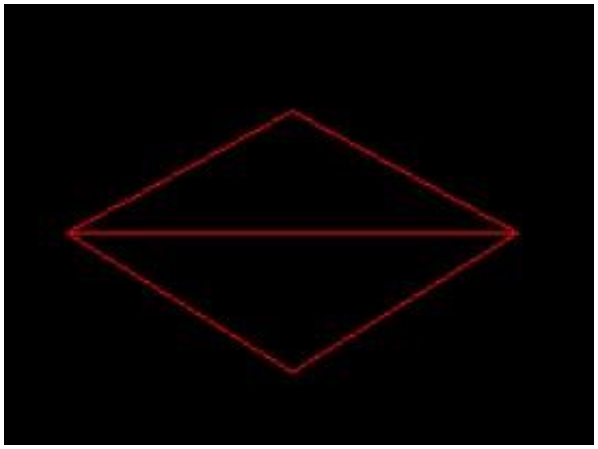

Figura 7 - plane 1 1

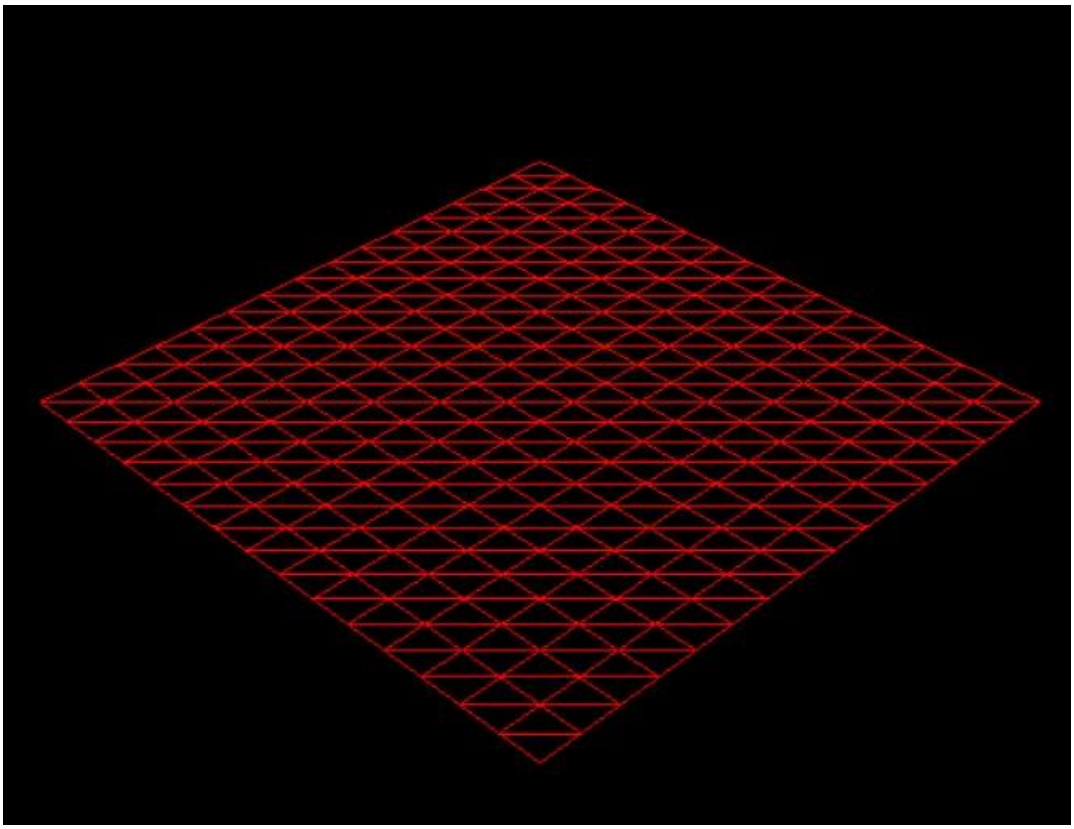

Figura 8 - plane 3 5

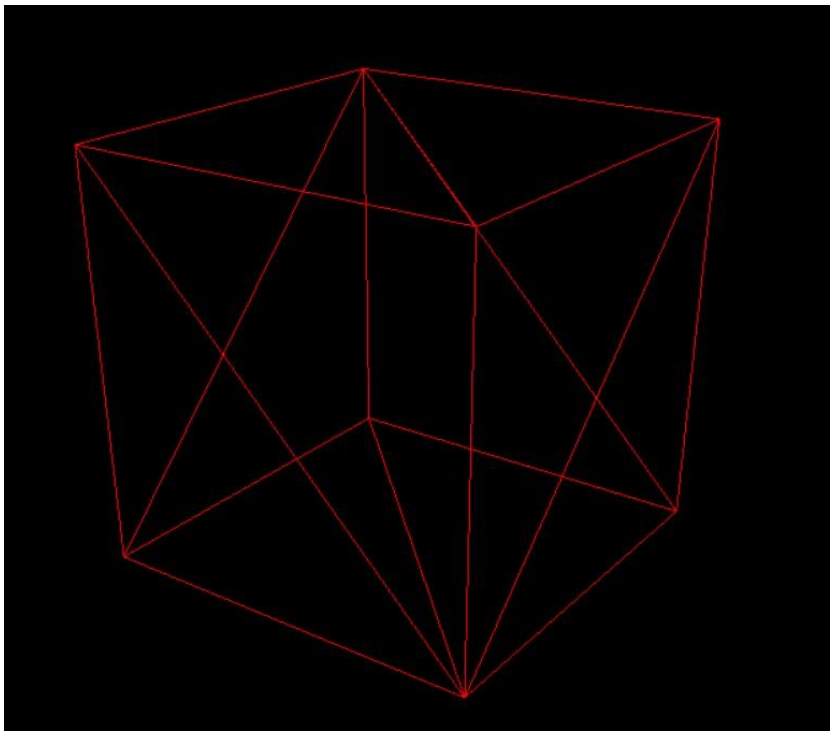

Figura 9 - box 1 1

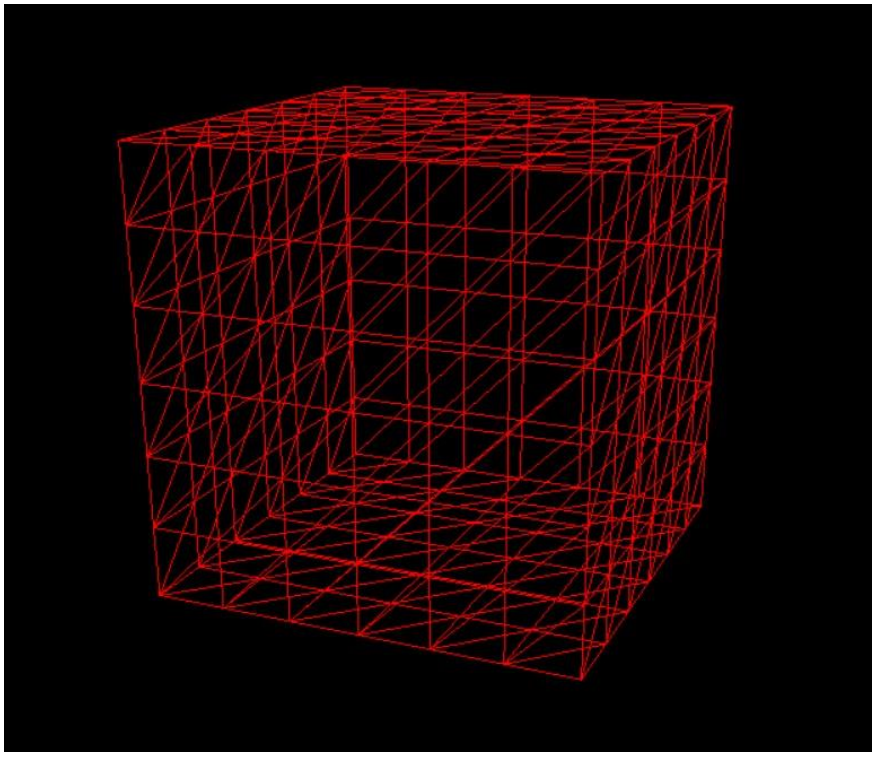

Figura 10 - box 5 6

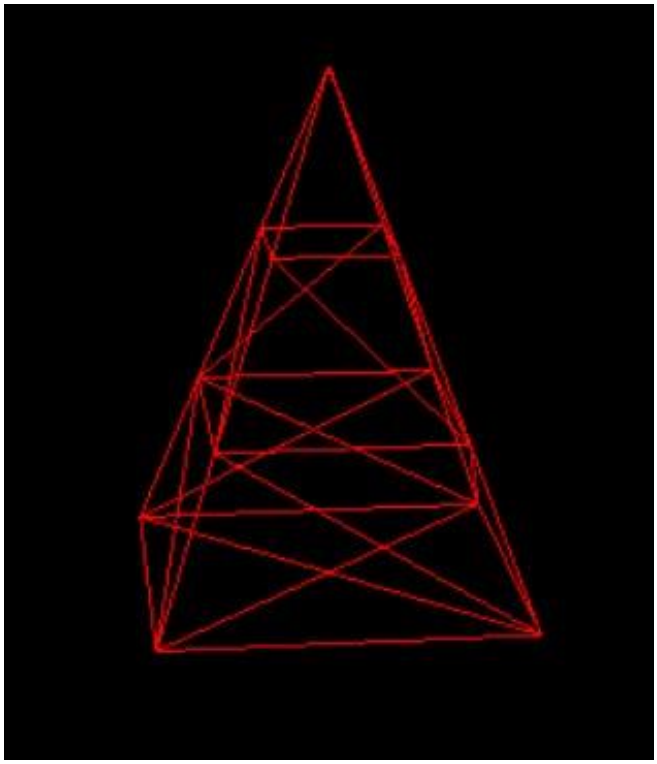

Figura 11 - cone 1 2 4 3

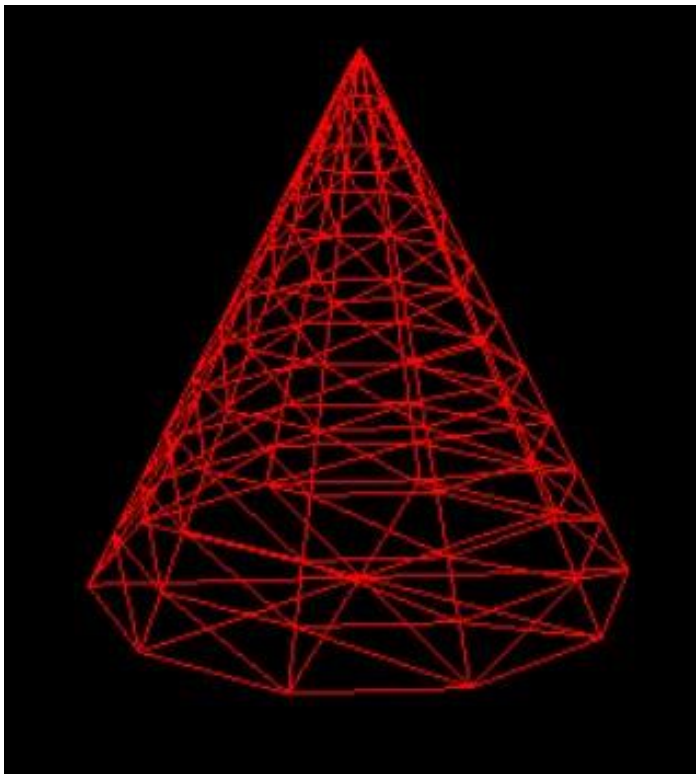

Figura 12 - cone 1 2 10 10

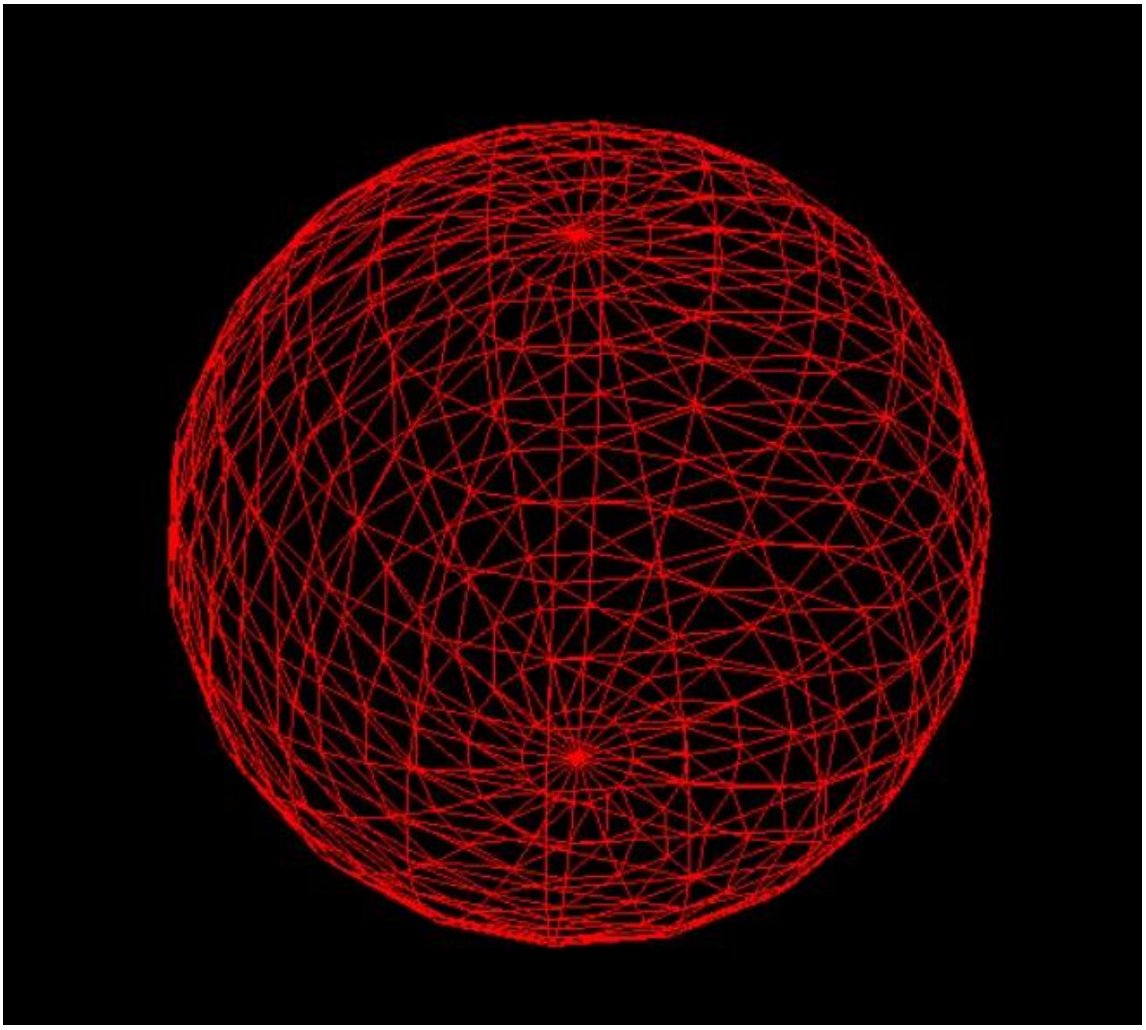

Figura 13 - sphere 1 20 20