INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ (IFCE) – CAMPUS FORTALEZA

DEPARTAMENTO DE TELEMÁTICA

CURSO: ENGENHARIA DE COMPUTAÇÃO

DISCIPLINA: COMPUTAÇÃO GRÁFICA 2021.1

PROFESSOR: AJALMAR REGO DA ROCHA NETO, DR

ALUNOS:

ELYSSON GABRIEL SOARES SIMÕES

JOÃO MARCUS MAIA ROCHA

RELATÓRIO COM ANÁLISE DOS RESULTADOS OBTIDOS NA MODELAGEM DE SÓLIDOS

FORTALEZA – CE

JULHO/2021

SUMÁRIO

1. INTRODUÇÃO

Este relatório tem como objetivo mostrar a análise dos resultados obtidos na modelagem de sólidos. O trabalho foi desenvolvido utilizando a linguagem python na versão 3.8.10, para o plot das imagens, foi utilizado a biblioteca matplotlib na versão 3.4.1, e para a manipulação das matrizes, foi utilizado a biblioteca Numpy versão 1.20.2. O algoritmo foi baseado nas aulas de Computação Gráfica do primeiro semestre de 2021, ministradas pelo professor Ajalmar Rego da Rocha Neto e no livro *Computação Gráfica: teoria e prática: geração de imagens / Aura Conci, Cristina Nader Vasconcelos, Eduardo Azevedo. - 2. ed. - Rio de Janeiro: Elsevier, 2018.*

2. DESENVOLVIMENTO

2.1. MODELAGEM DOS SÓLIDOS

Os objetos foram construídos de maneira que cada um estivesse descrito em termos de seu próprio Sistema de Coordenadas do Objeto (SCO). Para representar esses sólidos, foram utilizadas uma lista de vértices e uma lista de arestas. A lista de vértices é um vetor associativo (ou dicionário em python) em que a chave é o nome de um vértice e o valor é uma lista com a posição do vértice no sistema de coordenadas no formato [x, y, z]. A lista de aresta é um vetor associativo em que a chave é o nome da aresta e o valor uma lista com os vértices pertencentes à aresta. Os seguintes sólidos foram modelados:

- Cubo de lado igual a 1.5, com origem no centro do quadrado inferior do cubo e aresta do quadrado inferior paralela ao eixo x.

- Paralelepípedo com lados iguais a 1.5 em x, 5.0 em y e 2.5 em z, com origem em um dos vértices pertencentes ao retângulo inferior e aresta paralela ao eixo y.

- Pirâmide com base quadrada de lado igual a 2.0 e altura igual a 3.0, com origem no centro do quadrado da pirâmide e de tal maneira que uma aresta do quadrado faça ângulo de 45 graus com o eixo x.

- Tronco de pirâmide com bases quadradas de lados, respectivamente, iguais a 3.0 e 1.3, com altura de 2.5.

2.1.1. CUBO

Vértices = {'V1': [-0.75, -0.75, 0], 'V2': [0.75, -0.75, 0], 'V3': [-0.75, 0.75, 0], 'V4': [-0.75, -0.75, 1.5], 'V5': [0.75, 0.75, 0], 'V6': [0.75, -0.75, 1.5], 'V7': [-0.75, 0.75, 1.5], 'V8': [0.75, 0.75, 1.5]}

Arestas = {'A1': ('V1', 'V2'), 'A2': ('V1', 'V3'), 'A3': ('V1', 'V4'), 'A4': ('V2', 'V5'), 'A5': ('V2', 'V6'), 'A6': ('V3', 'V5'), 'A7': ('V3', 'V7'), 'A8': ('V4', 'V6'), 'A9': ('V4', 'V7'), 'A10': ('V5', 'V8'), 'A11': ('V6', 'V8'), 'A12': ('V7', 'V8')}

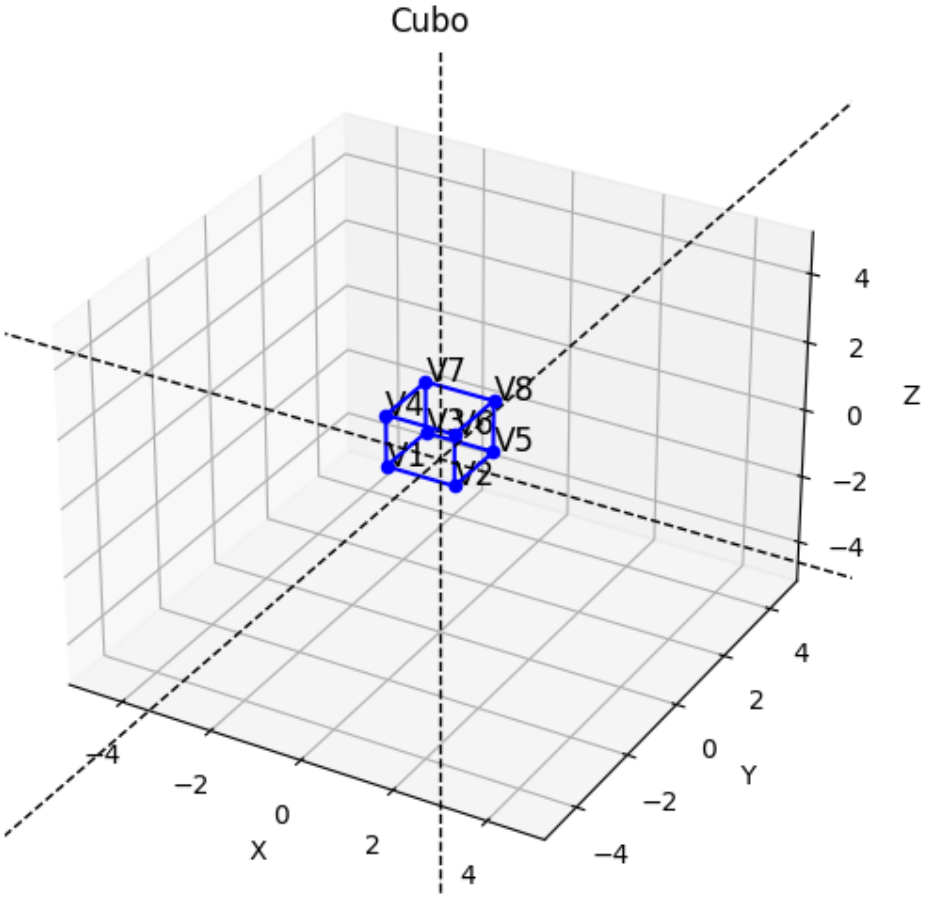

Figura 1: Modelagem do cubo

2.1.2. PARALELEPÍPEDO

Vértices = {'V1': [0, 0, 0], 'V2': [1.5, 0, 0], 'V3': [0, 0, 2.5], 'V4': [0, 5, 0], 'V5': [1.5, 0, 2.5], 'V6': [1.5, 5, 0], 'V7': [0, 5, 2.5], 'V8': [1.5, 5, 2.5]}

Arestas = {'A1': ('V1', 'V2'), 'A2': ('V1', 'V3'), 'A3': ('V1', 'V4'), 'A4': ('V2', 'V5'), 'A5': ('V2', 'V6'), 'A6': ('V3', 'V5'), 'A7': ('V3', 'V7'), 'A8': ('V4', 'V6'), 'A9': ('V4', 'V7'), 'A10': ('V5', 'V8'), 'A11': ('V6', 'V8'), 'A12': ('V7', 'V8')}

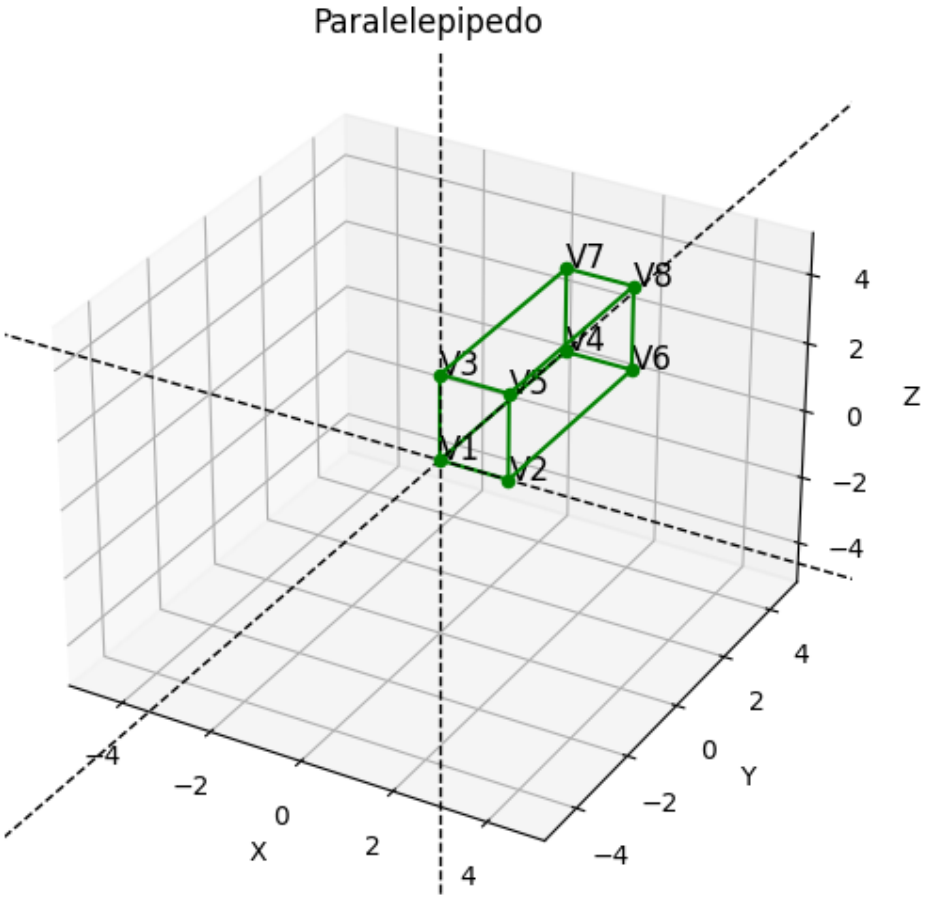

Figura 2: Modelagem do paralelepípedo

2.1.3. PIRÂMIDE

Vértices = {'V1': [0, 0, 3], 'V2': [-1.4142135623730951, 0, 0], 'V3': [0, 1.4142135623730951, 0], 'V4': [1.4142135623730951, 0, 0], 'V5': [0, -1.4142135623730951, 0]}

Arestas = {'A1': ('V1', 'V2'), 'A2': ('V1', 'V3'), 'A3': ('V1', 'V4'), 'A4': ('V1', 'V5'), 'A5': ('V2', 'V3'), 'A6': ('V2', 'V5'), 'A7': ('V3', 'V4'), 'A8': ('V4', 'V5')}

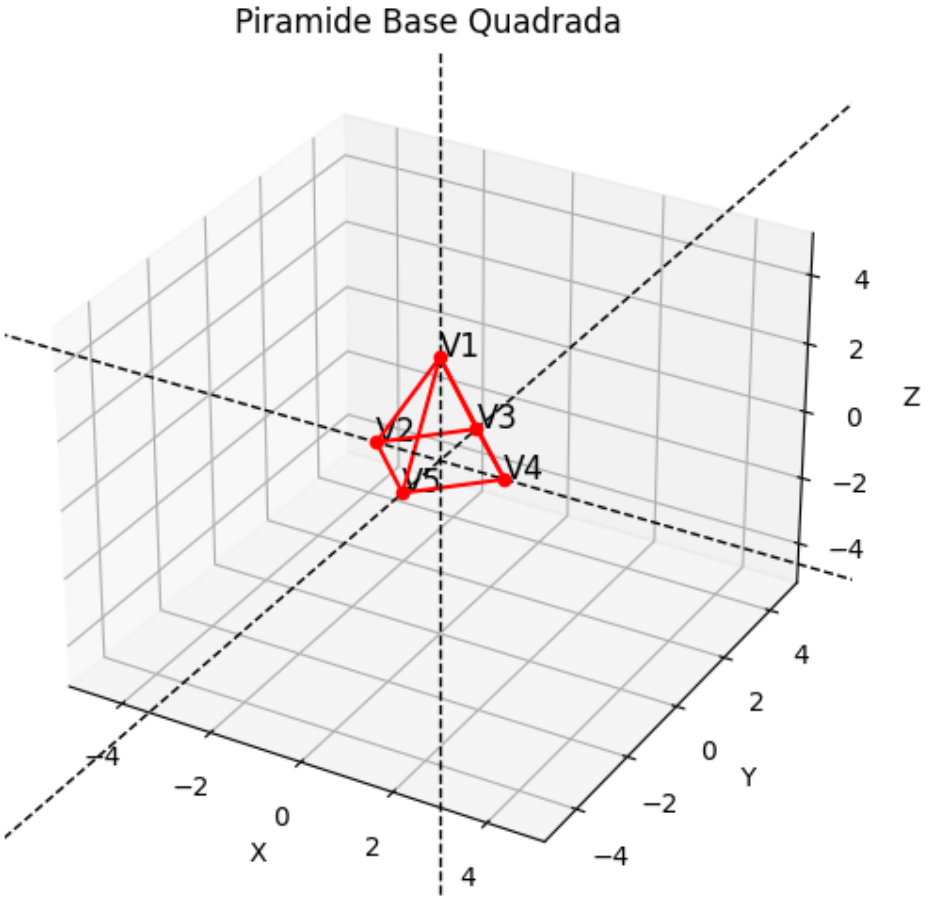

Figura 3: Modelagem da pirâmide

2.1.4. TRONCO DE PIRÂMIDE

Vértices = {'V1': [-1.140175425099138, 0, 2.5], 'V2': [-1.7320508075688772, 0, 0], 'V3': [0, 1.7320508075688772, 0], 'V4': [1.7320508075688772, 0, 0], 'V5': [0, -1.7320508075688772, 0], 'V6': [0, -1.140175425099138, 2.5], 'V7': [1.140175425099138, 0, 2.5], 'V8': [0, 1.140175425099138, 2.5]}

Arestas = {'A1': ('V1', 'V2'), 'A2': ('V1', 'V6'), 'A3': ('V1', 'V8'), 'A4': ('V3', 'V8'), 'A5': ('V2', 'V3'), 'A6': ('V2', 'V5'), 'A7': ('V3', 'V4'), 'A8': ('V4', 'V5'), 'A9': ('V4', 'V7'), 'A10': ('V5', 'V6'), 'A11': ('V6', 'V7'), 'A12': ('V7', 'V8')}

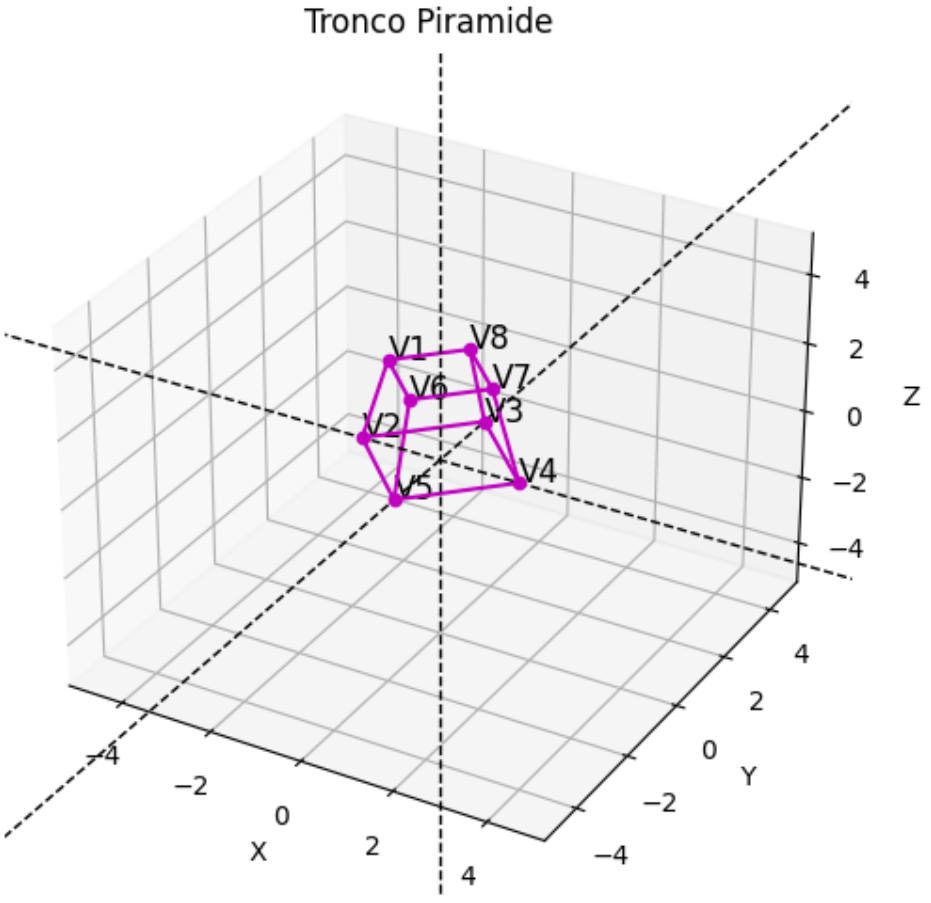

Figura 4: Modelagem do tronco de pirâmide

2.2. OBJETOS NO SISTEMA DE COORDENADAS DO MUNDO

A cena ou cenário, seja 2D ou 3D, é formado pela composição de objetos posicionados e ajustados em relação uns aos outros em um mundo virtual. Para suportar tal composição, é usado um Sistema de Coordenadas do Mundo (SCM), pois é necessário estabelecer uma base de referência única ou universal.

Utilizamos transformações, como translação e rotação, para inserir um objeto que está modelado no Sistema de Coordenadas do Objeto no Sistema de Coordenadas do Mundo.

A composição dos objetos, no sistema de coordenadas do mundo, respeitou as seguintes condições:

a. O cubo e a pirâmide devem estar localizados em apenas um octante do espaço, bem como o paralelepípedo e o tronco devem estar em apenas um octante. Além disso, pelo menos dois octantes adjacentes devem possuir sólidos.
b. O maior valor possível para cada uma das componentes de um vértice é 6. Se necessário aplique transformações de escala para que os sólidos sejam localizados respeitando tais limites.

Inserimos o cubo e a pirâmide no octante I, ou seja, $x > 0$, $y > 0$ e $z > 0$ para todos os pontos dos objetos. Já o paralelepípedo e o tronco foram colocados no octante II, ou seja, $x < 0$, $y > 0$ e $z > 0$ para todos os pontos dos objetos.

A seguir vemos uma imagem dos sólidos no Sistema de Coordenadas do Mundo:

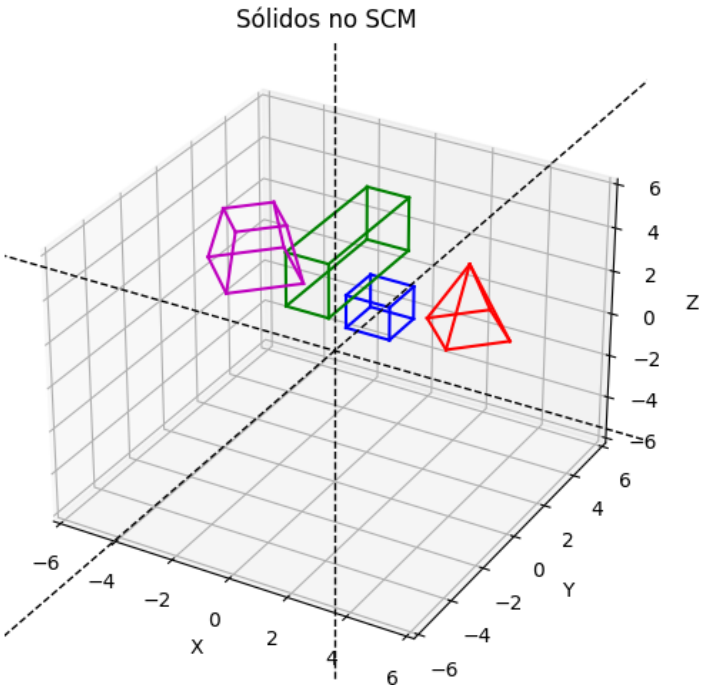

Figura 5: Sólidos no Sistema de Coordenadas do Mundo

2.3. OBJETOS NO SISTEMA DE COORDENADAS DA CÂMERA

A câmera virtual define um ponto de vista de uma cena e cria uma representação desta cena para o observador. Ela possui uma localização e orientação no Sistema de Coordenadas do Mundo.

Ao formar a imagem, consideramos apenas o que é visto pela câmera, ou seja, tudo aquilo que está dentro do seu volume de visão.

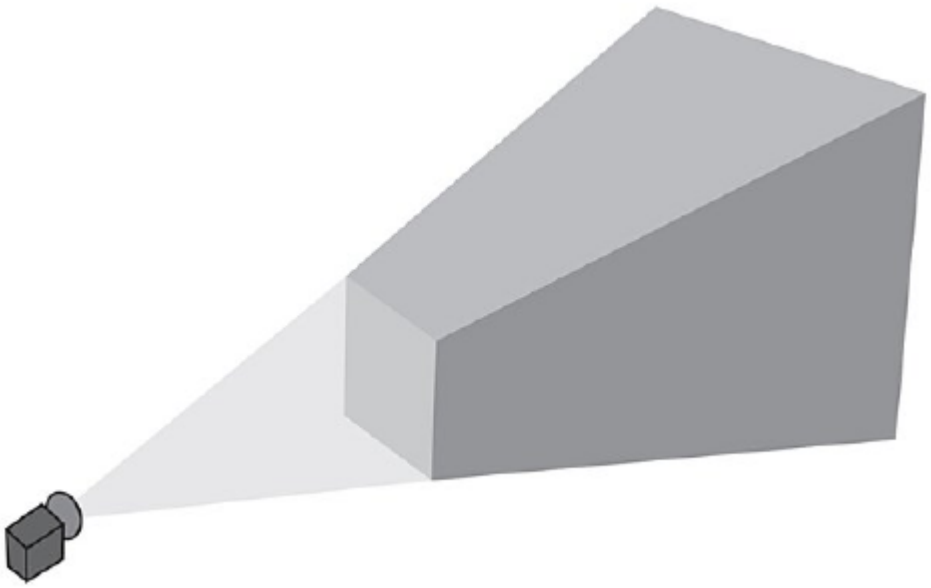

Figura 6: Exemplo de volume de visão

Ao fazer essa observação, criamos um novo sistema de coordenadas para facilitar a seleção dos objetos dentro do volume de visão e as projeções da cena 3D em 2D. O Sistema de Coordenadas da Câmera (SCC) tem como ponto de origem a posição da câmera no Sistema de Coordenadas do Mundo. Alinha-se o eixo z com a profundidade, o eixo x apontando para a direita da câmera e o eixo y para cima da câmera.

Os parâmetros extrínsecos são aqueles que descrevem como a câmera está posicionada no mundo virtual, ou seja, no Sistema de Coordenadas do Mundo. São eles a posição *p* da câmera, onde *p* = (x, y, z), o vetor $\vec{n}$, que determina a direção normal ao plano de projeção onde a imagem será formada, o vetor $\vec{v}$, que indica a direção vertical da imagem e é ortogonal a $\vec{n}$, e o vetor $\vec{u}$, que é ortogonal à $\vec{n}$ e . Os vetores $\vec{n}$, $\vec{v}$, e $\vec{u}$ devem ser normalizados.

Para posicionar a câmera em relação à cena, utilizamos o seguinte algoritmo:

1. Faça *eye* = (x_{eye}, y_{eye}, z_{eye}), a posição do observador, no caso, a câmera.
2. Faça *at* = (x_{at}, y_{at}, z_{at}), a posição para onde a câmera(*eye*) está olhando, no caso, este ponto será o centro de massa dos sólidos.
3. Um vetor up, que indica a direção pra cima de uma cena 3D, no caso, .
4. Faça $\vec{n}$ = *at* - *eye* e normalize-o. Então, $\vec{u}$ = (x_n, y_n, z_n).
5. Para obtermos $\vec{v}$, subtraímos de up sua projeção em $\vec{n}$, vide a fórmula na figura 6. Em seguida, normalizamos o vetor $\vec{v}$. Então, $\vec{v}$ = (x_v, y_v, z_v).
6. Para obtermos $\vec{u}$, basta fazer o produto vetorial entre $\vec{v}$ e $\vec{n}$, assim $\vec{u}$ já está normalizado. Então, $\vec{u}$ = (x_u, y_u, z_u).
7. Para levar o ponto de origem da câmera para a origem do novo sistema de coordenadas, devemos usar uma matriz de translação T, vide figura 7.
8. Para reorientar os eixos no novo sistema, devemos usar uma matriz de rotação R, vide figura 8.
9. Combinando a matriz de translação T e a matriz de rotação R, temos a matriz de transformação completa do Sistema de Coordenadas do Mundo para o Sistema de Coordenadas da Câmera, vide figura 10.

$$\vec{v} = \overrightarrow{up} - \frac{\overrightarrow{up} \cdot \vec{n}}{\vec{n} \cdot \vec{n}}$$

Figura 7: Fórmula para determinar o vetor v.

$$T = \begin{bmatrix} 1 & 0 & 0 & -x_c \\ 0 & 1 & 0 & -y_c \\ 0 & 0 & 1 & -z_c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figura 8: Matriz de translação que leva a câmera na posição (xc, yc, zc) para a origem.

$$R = \begin{bmatrix} x_u & y_u & z_u & 0 \\ x_v & y_v & z_v & 0 \\ x_n & y_n & z_n & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figura 9: A matriz de rotação que reorienta os novos eixos. Queremos transformar o vetor u como a direção horizontal, o vetor v como direção vertical e o vetor n como a direção de profundidade.

$$V = RT = \begin{bmatrix} x_u & y_u & z_u & -x_c * x_u - y_c * y_u - z_c * z_u \\ x_v & y_v & z_v & -x_c * x_v - y_c * y_v - z_c * z_v \\ x_n & y_n & z_n & -x_c * x_n - y_c * y_n - z_c * z_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figura 10: A matriz V é a combinação da matriz de rotação com a matriz de translação.

Escolhemos um dos octantes sem nenhum sólido, no caso o octante IV, ou seja x > 0, y < 0 e z > 0. E escolhemos um ponto como origem para o sistema de coordenadas da câmera, no caso o ponto (5, 4, -1). Utilizamos o ponto médio entre os centros de massa para ser o ponto para onde a câmera está olhando, ou seja $\vec{at}$.

O vetor up utilizado foi (0, 1, 0).

Executamos o algoritmo e transformamos os objetos do sistema de coordenadas do mundo para o sistema de coordenadas da câmera. Para verificarmos se a transformação foi feita corretamente, verificamos se a distância entre o vértice V1 do cubo até a origem da câmera (eye) no Sistema de Coordenadas do Mundo, é igual à distância entre o vértice V1 do cubo e a origem do Sistema de Coordenadas da Câmera. A figura 10 mostra o resultado.

```
Tamanho do vetor diferença entre o vértice V1 do cubo e a origem da câmera (eye): 6.373774391990981
Tamanho do vetor diferença entre o vértice V1 do cubo e a origem das coordenadas do SCC (0, 0, 0): 6.373774391990981
```

Figura 11: Verificando a corretude da transformação do Sistema de Coordenadas do Mundo para o Sistema de Coordenadas da Câmera.

Segue imagem dos sólidos após a transformação:

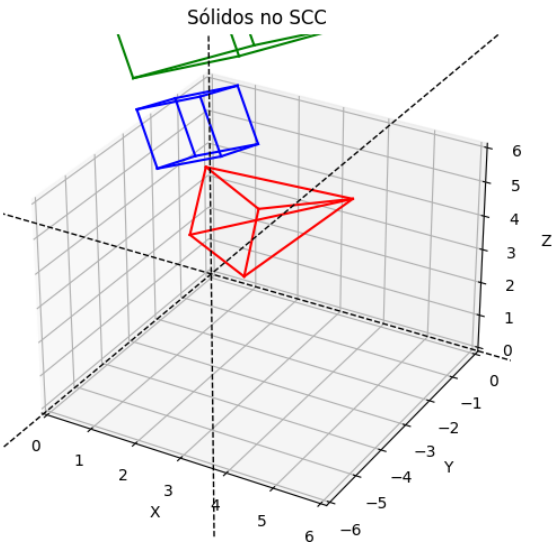

Figura 12: Sólidos no Sistema de Coordenadas da Câmera. Observe que alguns sólidos estão fora do volume de visão.

2.4. PROJEÇÃO DOS SÓLIDOS EM 2D

Nas projeções classificadas como paralelas, o centro de projeção é localizado no infinito e todas as linhas de projeção são paralelas entre si.

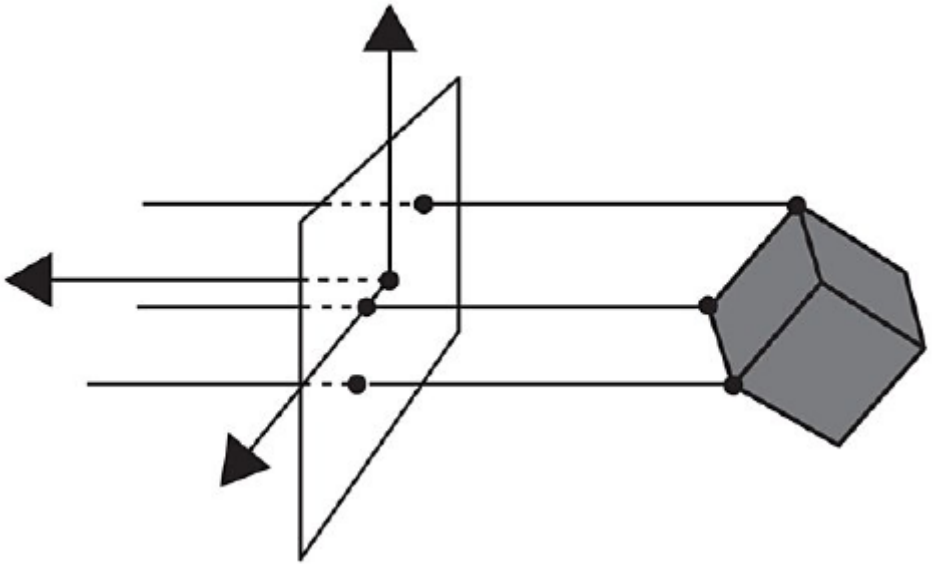

Figura 13: Projeção paralela ortogonal

A projeção paralela ortogonal é representada pela matriz P, vide figura.

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figura 14: Matriz da projeção paralela ortogonal.

Fizemos uma transformação de projeção paralela ortogonal dos sólidos contidos no volume de visão e projetamos as arestas dos sólidos em 2 dimensões, no caso usamos os eixos x e y.

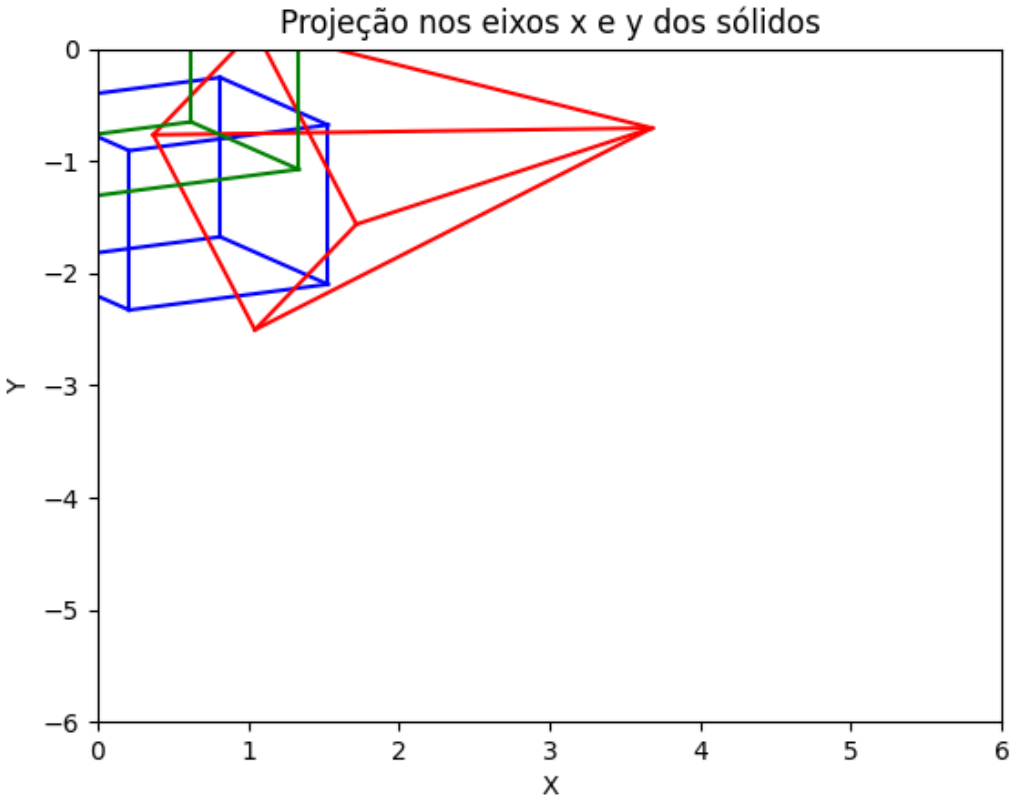

Figura 15: Resultado da projeção paralela ortogonal. Devemos obedecer o volume de visão.

3. CONCLUSÃO

Ao modelar os sólidos, e construí-los no seu próprio Sistema de Coordenadas do Objeto (SCO), podemos perceber que é possível representá-los graficamente utilizando uma lista de vértices e uma lista de arestas, apenas organizando essas estruturas em uma cena. Dessa forma, foi possível compor os diversos sólidos apresentados no desenvolvimento deste relatório em um único cenário, apenas tomando o cuidado de converter suas coordenadas para um Sistema de Coordenadas do Mundo (SCM).

É interessante ressaltar que ao escolher um ponto dentro um dos octantes do SCM, podemos considerá-lo como origem e assim computar a base vetorial para o Sistema de Coordenadas da Câmera (SCC). Com isso, podemos considerar um volume de visão através da nova origem e do ponto médio entre os centros de massas de cada um dos sólidos na derivação da nova base vetorial e assim transformando os objetos do SCM para o SCC.

Uma consequência da transformação citada anteriormente, é a possibilidade de realizar uma projeção paralela ortogonal que está dentro do volume de visão. Dessa forma podemos descrever a visualização a partir desse determinado ponto em duas dimensões.

4. REFERÊNCIAS BIBLIOGRÁFICAS

Aura Conci, Cristina Nader Vasconcelos, Eduardo Azevedo. Computação Gráfica: teoria e prática: geração de imagens. - 2. ed. - Rio de Janeiro: Elsevier, 2018.

Aulas de Computação Gráfica, professor ROCHA NETO, A. R. Ministradas no primeiro semestre de 2021, IFCE - Campus Fortaleza.

Matplotlib: Python plotting — Matplotlib 3.4.1 documentation. Disponível em: <https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.imshow.html>. Acesso em: 28 de jun. de 2021.