Universidade Estadual de Maringá

Departamento de Informática

Curso de Ciência da Computação

Disciplina – Computação Gráfica

Prof. Dr. Dante Alves Medeiros Filho

Trabalho de Implementação – Entrega/Apresentação: até 10/07/19

1)..Enunciado:

Programe um sistema de visualização baseado em projeção linear que contemple a projeção cilíndrica e cônica. Considere que são conhecidos os seguintes dados:

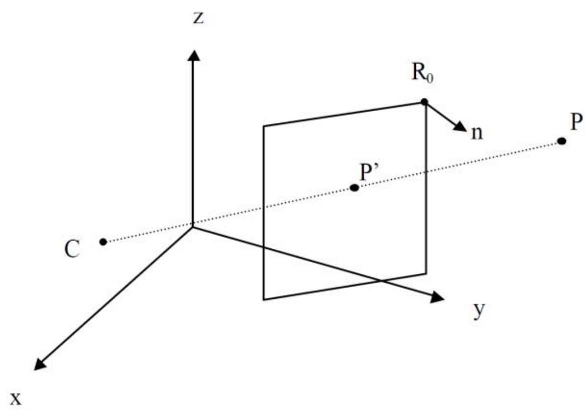

Figura-01 - Projeção Perspectiva

1.1) Ponto de Vista C(a,b,c): em coordenadas cartesianas **(a,b,c)** expressas no Sistema Global de Coordenadas ou Sistema Coordenadas do Mundo (**WCS**).

- **a** distância em x
- **b** distância em y
- **c** distância em z

1.2) Plano de Projeção - Sistema Global de Coordenadas

São fornecidos três pontos distintos e não colineares para definição do plano de projeção e um ponto sobre o plano $R_0 = (x_0, y_0, z_0)$:

$$\boldsymbol{P_1 = (x_1, y_1, z_1)}$$

$$\boldsymbol{P_2 = (x_2, y_2, z_2)}$$

$$\boldsymbol{P_3 = (x_3, y_3, z_3)}$$

Um ponto sobre o plano $\boldsymbol{R_0 = (x_0, y_0, z_0)}$ ou um dos pontos $\boldsymbol{P_1}$, $\boldsymbol{P_2}$, $\boldsymbol{P_3}$.

1.3) Dados do Objeto - Sistema Global de Coordenadas

$\boldsymbol{NV}$	Número de Vértices
$\boldsymbol{x_i, y_i, z_i}$	Coordenadas dos Vértices
$\boldsymbol{NS}$	Número de Superfícies
$\boldsymbol{NVPS_i}$	Número de Vértices por Superfície
$\boldsymbol{V_i}$	Vértices de uma determinada superfície – regra da mão direita

2) Cálculos:

2.1) Determinação do **Vetor Normal** ao Plano $\boldsymbol{\vec{n} = [n_x, n_y, n_z]}$ utilizando os três pontos fornecidos $\boldsymbol{(P_1,\ P_2,\ P_3)}$:

Produto Vetorial:

$$\vec{n} = P_2P_1\ \ X\ \ P_2P_3$$

$$\vec{n} = n_x\,,\ n_y\,,\ n_z$$

$$P_2 - P_1 = (x_1 - x_2), (y_1 - y_2)\,, (z_1 - z_2)$$

$$P_2 - P_3 = (x_3 - x_2), (y_3 - y_2)\,, (z_3 - z_2)$$

$$\vec{n} = n_x\vec{\imath} + n_y\vec{\jmath} + n_z\vec{k}$$

$$n = \begin{pmatrix} \vec{i} & \vec{j} & \vec{k} \\ x_1 - x_2 & y_1 - y_2 & z_1 - z_2 \\ x_3 - x_2 & y_3 - y_2 & z_3 - z_2 \end{pmatrix}$$

$$\boldsymbol{n_x} = (y_1 - y_2)\,.\,(z_3 - z_2) - (y_3 - y_2)\,.\,(z_1 - z_2)$$

$$\boldsymbol{n_y} = -(x_1 - x_2)\,.\,(z_3 - z_2) - (x_3 - x_2)\,.\,(z_1 - z_2)$$

$$\boldsymbol{n_z} = (x_1 - x_2)\,.\,(y_3 - y_2) - (x_3 - x_2)\,.\,(y_1 - y_2)$$

2.2) Calculo de , $\boldsymbol{d_0}$, $\boldsymbol{d_1}$ e $\boldsymbol{d}$:

$$d_0 = x_0\,.\,n_x + y_0.\,n_y + z_0\,.\,n_z$$

$$d_1 = a\,.\,n_x + b.\,n_y + c\,.\,n_z$$

$$d = d_0 - d_1$$

2.3) Calculo da Matriz Perspectiva

$$M_{per} = \begin{pmatrix} d + an_x & an_y & an_z & -ad_0 \\ bn_x & d + bn_y & bn_z & -bd_0 \\ cn_x & cn_y & d + cn_z & -cd_0 \\ n_x & n_y & n_z & -d_1 \end{pmatrix}$$

2.4) Calculo das Coordenadas no Plano de Projeção:

$$P' = M_{per}\,.\,M_{objeto}$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ w' \end{pmatrix} = \begin{pmatrix} d + an_x & an_y & an_z & -ad_0 \\ bn_x & d + bn_y & bn_z & -bd_0 \\ cn_x & cn_y & d + cn_z & -cd_0 \\ n_x & n_y & n_z & -d_1 \end{pmatrix} . M_{objeto}$$

Resultados em Coordenadas homogêneas:

$$P' = (x', y', z', w')$$

Transformar em Coordenadas Cartesianas:

$$x_c = \frac{x'}{w'}$$

$$y_c = \frac{y'}{w'}$$

$$z_c = \frac{z'}{w'}$$

$$1 = \frac{w}{w'}$$

Transformar em Coordenadas do Plano:

$$x_p = x_c$$

$$y_p = y_c$$

Transformar em Coordenadas do Dispositivo. É uma Transformação Janela-Viewport:

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = T_{j-v} \begin{bmatrix} x_p \\ y_p \end{bmatrix}$$

Observações: o sistema deverá prover a translação em 3D dos objetos da cena e o posicionamento dinâmico do plano de projeção e ponto de vista. Além disso, deverá levar em consideração a possível diferença de ***aspect ratio*** e a centralização da projeção quando da transformação **Janela-Viewport**.

Quando as razões de aspecto forem iguais:

$$T_{janela-viewport} = \begin{bmatrix} S_x & 0 & u_{min} - S_x x_{min} \\ 0 & -S_y & S_y y_{max} + v_{min} \\ 0 & 0 & 1 \end{bmatrix}$$

Caso sejam diferentes:

if $\mathrm{R_w} > \mathrm{R_v}$ then

$$\mathrm{u_{máxnovo}} = \mathrm{R_w}(\mathrm{v_{máx}} - \mathrm{v_{min}}) + \mathrm{u_{min}}$$

$$\begin{bmatrix} \mathrm{u} \\ \mathrm{v} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathrm{s_x} & 0 & -\mathrm{S_x x_{min}} + \dfrac{\mathrm{u_{máx}}}{2} - \dfrac{\mathrm{u_{máxnovo}}}{2} + \mathrm{u_{min}} \\ 0 & -\mathrm{s_y} & \mathrm{S_y y_{max}} + \mathrm{v_{min}} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ 1 \end{bmatrix}$$

Else

$$\mathrm{v_{máxnovo}} = \frac{(\mathrm{u_{máx}} - \mathrm{u_{min}})}{\mathrm{R_w}} + \mathrm{v_{min}}$$

$$\begin{bmatrix} \mathrm{u} \\ \mathrm{v} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathrm{s_x} & 0 & \mathrm{u_{min}} - \mathrm{S_x x_{min}} \\ 0 & -\mathrm{s_y} & \mathrm{S_y y_{max}} + \dfrac{\mathrm{v_{máx}}}{2} - \dfrac{\mathrm{v_{máxnovo}}}{2} + \mathrm{v_{min}} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \\ 1 \end{bmatrix}$$

Figura 02: Pseudocódigo para cálculo de novos valores para os limites para mapeamento