UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACULDADE DE COMPUTAÇÃO

NOTAS DE AULA DE

COMPUTAÇÃO GRÁFICA

PAULO PAGLIOSA

CAMPO GRANDE, MS

CAPÍTULO 5

Câmera Virtual

5.1 Introdução

A câmera virtual define a porção do espaço que é visível por um observador, chamada *volume de vista*. A parte da cena no interior do volume de vista e mais próxima do observador será projetada na janela de projeção. A forma do volume de vista depende do *tipo de projeção* considerado. Para projeção perspectiva, o volume de vista, derivado diretamente do modelo de câmera *buraquinho de alfinete* visto no Capítulo 1, é um tronco de pirâmide denominado *frustum*. Além da projeção perspectiva, é considerada também nesse capítulo a projeção paralela ortográfica, cujo volume de vista é um paralelepípedo. Os parâmetros de configuração da câmera virtual utilizados para especificação de um volume de vista arbitrário no espaço de coordenadas global (WC) são apresentados na Seção 5.2.

A determinação da interseção de um volume de vista com os atores de uma cena e a determinação de quais superfícies são visíveis ou escondidas dependem do método de síntese de imagens adotado. Na OpenGL, por exemplo, a interseção é baseada em algoritmos de *recorte* de pontos, linhas e polígonos, enquanto os pontos mais próximos do observador dependem de testes de profundidade conforme descrito no Capítulo 6. Em geral, esses algoritmos podem ser mais fácil e eficientemente implementados para volumes de vista com forma, posição e orientação pré-definidas, chamados *volumes de vista canônicos*. A Seção 5.2 mostra como transformar as coordenadas de um ponto em relação ao sistema global para coordenadas do espaço do volume de vista canônico da OpenGL, denominado NDC (*normalized device coordinates*). Já no traçado de raios, a remoção das superfícies escondidas é efetuada pela interseção de raios com os atores da cena, como visto no Capítulo 3. As equações dos raios de pixel para as projeções perspectiva e paralela são deduzidas na Seção 5.4.

Após a projeção de um ponto visível na janela de projeção, este deve ter suas coordenadas transformadas para coordenadas de imagem. O mapeamento janela de projeção para imagem e o mapeamento inverso imagem para janela de projeção são tratados na Seção 5.5. Por fim, a Seção 5.6 relaciona algumas operações com a câmera.

5.2 Especificação de Volumes de Vista

A configuração da câmera virtual no espaço global, e em consequência o volume de vista por ela determinado, é definida pelos parâmetros ilustrados na Figura 5.1.

Figura 5.1: Parâmetros da câmera virtual.

O ponto de vista do observador é dado por um ponto C, a *posição* da câmera, com coordenadas tomadas em relação ao WC. O *sistema de coordenadas de vista*, ou *de câmera*, VRC (*view reference coordinates*), é um sistema Cartesiano com origem em C e eixos x_c, y_c e z_c cujas direções são dadas por versores $\mathbf{u}$, $\mathbf{v}$ e $\mathbf{n}$, respectivamente, todos com coordenadas em relação à base canônica do $\mathbb{R}^3$. O versor $\mathbf{n}$, também denotado por **VPN** (*view plane normal*), é perpendicular ao plano de projeção, ou plano de vista, o qual está situado a uma distância $F > 0$ da posição da câmera na direção inversa de $\mathbf{n}$. Os versores $\mathbf{u}$ e $\mathbf{v}$ fornecem a orientação da *janela de vista* ou *janela de projeção* (o "filme" do modelo de câmera buraquinho de alfinete), um retângulo de largura $W > 0$ e altura $H > 0$ no plano de projeção cuja *razão de aspecto* é definida como

$$A = \frac{W}{H}. \tag{5.1}$$

Para definir o VRC, usamos, além do ponto C, um ponto de referência $\mathrm{PF} \neq C$ para o qual a câmera aponta, chamado de *ponto focal*, com coordenadas tomadas em relação ao WC (note que o ponto focal não precisa estar situado no plano de projeção). A *direção de projeção* é dada pelo versor

$$\mathbf{DOP} = \frac{\mathrm{PF} - C}{D}, \tag{5.2}$$

em que $D = \|\mathrm{PF} - C\|$ é a distância entre a posição da câmera e o ponto focal. Ainda, usamos um *vetor "para cima"*, denotado por **VUP** (*view up vector*), com coordenadas em relação à base canônica, não colinear com **DOP**, tal que a projeção de **VUP** no plano de vista define a direção de $\mathbf{v}$. Valem as seguintes relações:

$$\begin{aligned} \mathbf{n} &= -\mathbf{DOP}, \\ \mathbf{u} &= \frac{\mathbf{VUP} \times \mathbf{n}}{\|\mathbf{VUP} \times \mathbf{n}\|}, \\ \mathbf{v} &= \frac{\mathbf{n} \times \mathbf{u}}{\|\mathbf{n} \times \mathbf{u}\|}. \end{aligned} \tag{5.3}$$

Como mencionado na Seção 5.1, a forma do volume de vista depende do tipo de projeção. Na projeção perspectiva, as faces laterais do volume de vista, chamado *frustum*, são definidas por projetores que passam pelas bordas da janela de vista e

convergem para a posição da câmera. A face da frente do *frustum* é a própria janela de projeção, enquanto a face de fundo, situada a uma distância B da posição do observador ao longo da direção de projeção, é um retângulo cujo plano é paralelo ao plano de vista, como ilustrado na Figura 5.2. Os planos contendo as faces de frente e de fundo são chamados de *planos de recorte* do volume de vista. Observe que o plano de recorte de frente é também o plano de projeção.

Figura 5.2: Volume de vista da projeção perspectiva.

Na projeção perspectiva, a altura da janela de vista pode ser definida em termos do *ângulo de vista vertical* θ. Da Figura 5.2, tem-se que

$$\tan\left(\frac{\theta}{2}\right) = \frac{H}{2F}, \tag{5.4}$$

ou, inversamente,

$$H = 2F\tan\left(\frac{\theta}{2}\right). \tag{5.5}$$

Na projeção paralela ortográfica, considera-se que a posição da câmera esteja a uma distância infinita do plano de projeção na direção de **VPN**. Neste caso, os projetores tornam-se paralelos entre si e perpendiculares ao plano de projeção (daí o denominação paralela ortográfica). O volume de vista torna-se, assim, um paralelepípedo, como ilustrado na Figura 5.3.

Figura 5.3: Volume de vista da projeção paralela ortográfica.

5.3 Transformações de Vértices em OpenGL

A interface de programação de aplicação (API) OpenGL provê funções para desenho de pontos, linhas e polígonos. Assim, a geometria de qualquer objeto gráfico a ser renderizado com OpenGL, independente de sua complexidade, deve ser descrita em termos de pontos, linhas e/ou polígonos — tipicamente triângulos —, chamados *primitivos gráficos*. Um primitivo, por sua vez, é definido por um ou mais pontos no espaço $\mathbb{E}^3$ chamados de *vértices*. Portanto, um ponto é caracterizado por somente

um vértice, uma linha por dois vértices conectados por um segmento de reta, e um triângulo por três vértices não colineares conectados por três segmentos de reta, as arestas do triângulo. Além das coordenadas espaciais, um vértice pode conter outros *atributos*, tais como versor normal, cor, etc.

Conforme descrito no Capítulo 6, um dos estágios do processo, ou *pipeline*, de geração de imagens em OpenGL é o *processamento de vértices*. As funcionalidades implementadas nesse estágio dependem da aplicação, mas o processamento de vértices sempre deve determinar, para cada vértice da cena, as *coordenadas de recorte*, utilizadas pela OpenGL para determinar se o vértice é visível ou não. Esta operação — assim como as demais funcionalidades do processamento de vértices—, deve ser implementada pelo desenvolvedor da aplicação. Isto quer dizer que, no núcleo da OpenGL, não há uma implementação *default* do processamento de vértices que efetue sequer o que a própria OpenGL espera como resultado desse estágio do *pipeline*, isto é, as coordenadas de recorte dos vértices. Tal implementação deve ser provida através de um módulo que executa diretamente na unidade de processamento gráfico, ou GPU (*graphics processing unit*), chamado *shader de vértice*.

As coordenadas de recorte para um dado volume de vista são coordenadas homogêneas $[x_r, y_r, z_r, w_r] \in \mathbb{T}_3$ tais que, para um vértice visível — isto é, que está no interior do volume de vista —, as coordenadas Cartesianas equivalentes $(x_n, y_n, z_n) \in \mathbb{E}^3$,

$$(x_n, y_n, z_n) = (x_r/w_r, y_r/w_r, z_r/w_r), \tag{5.6}$$

chamadas *coordenadas de dispositivo normalizadas*, satisfazem $-1 \leq x_n, y_n, z_n \leq 1$. Portanto, as coordenadas de dispositivo normalizadas dos vértices visíveis da cena devem ser mapeadas para um cubo de lado 2 u.c. (unidades de comprimento) centrado na origem. Diferentemente de outros sistemas de coordenadas Cartesianas, os eixos desse sistema, conhecido como NDC na literatura de OpenGL, são orientados de acordo com a *regra da mão esquerda*: tomando-se a tela do computador como referência, por exemplo, se o eixo x_n aponta para a direita e o eixo y_n para cima, então o eixo z_c do NDC aponta para dentro da tela, como ilustrado na Figura 5.4.

Figura 5.4: Volume de vista canônico da OpenGL.

Veremos agora quais transformações devem ser aplicadas a um vértice a fim de se determinar suas coordenadas de recorte.

5.3.1 Transformação Modelo-Vista

A primeira transformação é denominada *modelo-vista* e definida por uma matriz homogênea $\mathbf{M}_{\text{mv}}$ chamada matriz modelo-vista. De fato, esta transformação é uma composição de uma transformação de modelo $\mathbf{M}_{\text{m}}$ e uma transformação de vista $\mathbf{M}_{\text{v}}$,

isto é: $\mathbf{M}_{\text{mv}} = \mathbf{M}_{\text{v}} \cdot \mathbf{M}_{\text{m}}$. A transformação de modelo é responsável por transformar as *coordenadas locais* $[x_o, y_o, z_o, 1]$ de um vértice de um objeto em *coordenadas globais* $[x, y, z, 1]$ da cena:

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{M}_{\text{m}} \cdot \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}.$$

Por exemplo, uma aplicação poderia definir uma caixa "padrão" como sendo um cubo de lado 2 u.c. centrado na origem de um sistema de coordenadas local da caixa; caixas com quaisquer tamanhos e orientações e com centros em quaisquer posições poderiam ser adicionadas à cena através de múltiplas *instâncias* da caixa "padrão", sendo que para cada instância seria definida uma transformação $\mathbf{M}_{\text{m}}$ levando as coordenadas dos vértices de seu sistema local para o sistema global da cena. A matriz $\mathbf{M}_{\text{m}}$ é usualmente uma composição de uma translação, rotação e transformação de escala (TRS), como discutido nos estudos do Capítulo 2.

A transformação de vista é responsável por transformar as coordenadas globais de um vértice para em coordenadas *coordenadas de vista*, ou *de câmera*, $[x_c, y_c, z_c, 1]$, isto é, tomadas em relação ao VRC:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \mathbf{M}_{\text{v}} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}.$$

Se considerarmos o VRC como um sistema local, a matriz de transformação de vista pode ser obtida, conforme visto no Capítulo 2, pela composição de uma translação da posição da câmera para a origem do sistema global seguida de uma rotação para alinhamento dos eixos do sistema de vista com os eixos do sistema global, isto é,

$$\mathbf{M}_{\text{v}} = \begin{bmatrix} u_x & u_y & u_z & -\mathbf{u} \cdot \mathbf{c} \\ v_x & v_y & v_z & -\mathbf{v} \cdot \mathbf{c} \\ n_x & n_y & n_z & -\mathbf{n} \cdot \mathbf{c} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.7}$$

em que $\mathbf{c} = C - O$ é o vetor da origem do sistema global à posição da câmera. Note que, após a transformação de vista, é como se a câmera estivesse posicionada na origem $O(0, 0, 0)$ e apontada na direção do eixo z negativo do sistema global.

5.3.2 Transformação de Projeção

A segunda transformação é a chamada *projeção* (embora não seja uma projeção propriamente dita), dada por uma matriz homogênea $\mathbf{M}_{\text{p}}$ que transforma as coordenadas de vista de um vértice em coordenadas de recorte:

$$\begin{bmatrix} x_r \\ y_r \\ z_r \\ w_r \end{bmatrix} = \mathbf{M}_{\text{p}} \cdot \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}. \tag{5.8}$$

$\mathbf{M}_{\mathrm{p}}$ depende da forma do volume de vista que, por sua vez, depende do tipo de projeção. Os volumes de vista da Seção 5.2 são simétricos em relação ao eixo z_c do sistema de vista, isto é, as coordenadas dos cantos inferior esquerdo e superior direito da janela de vista, em relação ao VRC, são $(-W/2.-H/2,-F)$ e $(W/2.H/2,-F)$, respectivamente. A seguir, deduziremos as matrizes de projeção para as projeções perspectiva e paralela considerando que os volumes de vista podem ser não simétricos em relação ao eixo z_c, uma vez que essas formas são comumente encontradas na literatura. Assim, ao invés de W e H, vamos usar os parâmetros l, r, b e t para especificar a posição e dimensões da janela de vista, tal que as coordenadas dos cantos inferior esquerdo e superior direito da janela de vista, em relação ao VRC, ficam $(l.b,-F)$ e $(r,t,-F)$, respectivamente, como ilustrado na Figura 5.5. Obtidas as genéricas, as matrizes para os casos particulares simétricos também são apresentadas.

Figura 5.5: Janela de vista não simétrica em relação ao eixo z_c.

Projeção Perspectiva

Para a projeção perspectiva, a face no plano de recorte de fundo de um *frustum* não simétrico está a uma profundidade $-F$ a partir da origem do sistema de vista, ao longo da direção de projeção, sendo as coordenadas, em relação ao sistema de vista, dos cantos inferior esquerdo e superior direito $(Bl/F, Bb/F, -B)$ e $(Br/F, Bt/F.-B)$, respectivamente (obtidas diretamente por semelhança de triângulos).

$\mathbf{M}_{\mathrm{p}}$ pode ser determinada como segue. Seja $(x_p, y_p, -F)$ a projeção do vértice (x_c, y_c, z_c), todas as coordenadas em relação ao sistema de vista. Por semelhança de triângulos, da Figura 5.6 tem-se que:

$$x_p = \frac{-F\,x_c}{z_c} = \frac{F\,x_c}{-z_c}. \tag{5.9}$$

Da mesma maneira,

$$y_p = \frac{-F\,y_c}{z_c} = \frac{F\,y_c}{-z_c}. \tag{5.10}$$

Figura 5.6: Projeção perspectiva de um ponto.

As coordenadas x_p e y_p não são ainda coordenadas de recorte, pois as coordenadas de recorte, após a divisão de perspectiva, devem resultar em coordenadas de dispositivo normalizadas, ou NDCs. Para tal, mapeia-se linearmente $l \leq x_p \leq r$ para $-1 \leq x_n \leq 1$, o que resulta na reta

$$x_n = \frac{2}{r-l}x_p + x_0.$$

Para $x_p = r$, tem-se $x_n = 1$; logo, $x_0 = -(r+l)/(r-l)$. Substituindo-se a Equação (5.9) na equação acima, obtém-se

$$x_n = \underbrace{\left(\frac{2n}{r-l}x_c + \frac{r+l}{r-l}z_c\right)}_{x_r} \Big/ - z_c. \tag{5.11}$$

Da mesma maneira, vamos mapear linearmente $b \leq y_p \leq t$ para $-1 \leq y_n \leq 1$:

$$y_n = \frac{2}{t-b}y_p + y_0.$$

Para $y_p = t$, tem-se $y_n = 1$; logo, $y_0 = -(t+b)/(t-b)$. Substituindo-se a Equação (5.10) na equação acima, obtém-se

$$y_n = \underbrace{\left(\frac{2F}{t-b}y_c + \frac{t+b}{t-b}z_c\right)}_{y_r} \Big/ - z_c. \tag{5.12}$$

Visto que $x_n = x_r/w_r$ e $y_n = y_r/w_r$, das Equações (5.11) e (5.12) tem-se que $w_r = -z_c$,

$$\begin{aligned} x_r =& \frac{2F}{r-l}x_c + \frac{r+l}{r-l}z_c, \text{ e} \\ y_r =& \frac{2F}{t-b}y_c + \frac{t+b}{t-b}z_c. \end{aligned}$$

Escrevendo-se matricialmente:

$$\begin{bmatrix} x_r \\ y_r \\ z_r \\ w_r \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{2F}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2F}{t-b} & \frac{t+b}{t-b} & 0 \\ \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & -1 & 0 \end{bmatrix}}_{\mathbf{M}_\mathrm{p}} \cdot \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}.$$

(Note que $\mathbf{M}_\mathrm{p}$ não é afim, ou seja, $w_r \neq 1$.)

Para determinar a terceira linha da matriz, é necessário determinar o valor de z_n (com o calcula-se z_r). Esse valor é usado pela OpenGL para o recorte e também para o teste de visibilidade (que leva em conta a profundidade do vértice em NDC, como

visto na Seção 5.5 e também no Capítulo 6). Uma vez que não se quer que $z_n = z_r/w_r$ dependa de x_c nem de y_c, pode-se escrever

$$\begin{bmatrix} x_r \\ y_r \\ z_r \\ w_r \end{bmatrix} = \begin{bmatrix} \frac{2F}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2F}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \alpha & \beta \\ 0 & 0 & -1 & 0 \end{bmatrix} . \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} .$$

Assim, com $w_r = -z_c$ e $z_r = \alpha z_c + \beta$, tem-se

$$z_n = \frac{\alpha z_c + \beta}{-z_c}.$$

Para $z_c = -F$ tem-se $z_n = -1$, enquanto para $z_c = -B$ tem-se $z_n = 1$. Escrevendo-se a relação acima para esses dois pares de valores obtém-se o sistema:

$$\begin{cases} -\alpha F + \beta = -F \\ -\alpha B + \beta = B \end{cases}$$

cuja solução fornece

$$\alpha = -\frac{B+F}{B-F}$$
$$\beta = -\frac{2BF}{B-F}.$$

Portanto, a matriz de projeção final para o *frustum* genérico é:

$$\mathbf{M}_{\mathrm{p}} = \begin{bmatrix} \frac{2F}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2F}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{B+F}{B-F} & -\frac{2BF}{B-F} \\ 0 & 0 & -1 & 0 \end{bmatrix} \tag{5.13}$$

Se o volume de vista for simétrico em relação a z_c, ou seja, $r = -l = W/2$ e $t = -b = H/2$, a matriz de projeção fica:

$$\mathbf{M}_{\mathrm{p}} = \begin{bmatrix} \frac{2F}{W} & 0 & 0 & 0 \\ 0 & \frac{2F}{H} & 0 & 0 \\ 0 & 0 & -\frac{B+F}{B-F} & -\frac{2BF}{B-F} \\ 0 & 0 & -1 & 0 \end{bmatrix} \tag{5.14}$$

Projeção Paralela

Para a projeção paralela, as coordenadas dos cantos inferior esquerdo e superior direito da face do paralelepípedo no plano de recorte de fundo, em relação ao VRC, são $(l, b, -B)$ e $(r, t, -B)$, respectivamente. A projeção paralela de um vértice (x_c, y_c, z_c) é $(x_c, y_c, -F)$.

$\mathbf{M}_{\text{p}}$ pode ser determinada fazendo-se interpolações lineares de todas as coordenadas no sistema de vista para NDC, como feito para x_r e y_r na projeção perspectiva. Alternativamente, a matriz pode ser obtida combinando-se:

1. uma translação do centro do volume de vista para a origem do sistema de vista (isto é, para a posição da câmera), e
2. uma transformação de escala em torno da origem do sistema de vista para que o volume de vista fique com o tamanho do cubo NDC:

$$\mathbf{M}_{\text{p}} = \mathbf{S}\left(\frac{2}{r-l}, \frac{2}{t-b}, -\frac{2}{B-F}\right) \cdot \mathbf{T}\left(-\frac{r+l}{2}, -\frac{t+b}{2}, \frac{B+F}{2}\right).$$

A matriz de projeção resultante é:

$$\mathbf{M}_{\text{p}} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{B-F} & -\frac{B+F}{B-F} \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.15}$$

Se o volume de vista for simétrico em relação a z_c ($r = -l = W/2$ e $t = -b = H/1$), a matriz de projeção paralela fica:

$$\mathbf{M}_{\text{p}} = \begin{bmatrix} \frac{2}{W} & 0 & 0 & 0 \\ 0 & \frac{2}{H} & 0 & 0 \\ 0 & 0 & -\frac{2}{B-F} & -\frac{B+F}{B-F} \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.16}$$

5.4 Câmera Virtual e Traçado de Raios

O traçado de raios usa a câmera virtual apenas para definir quais são a origem e direção do raio de pixel que passa pelo ponto P de coordenadas $(x_p, y_p, -F)$ em relação ao VRC, $-W/2 \leq x_p \leq W/2$ e $-H/2 \leq y_p \leq H/2$ (vamos considerar apenas volumes de vista simétricos em relação a z_c).

Para a projeção perspectiva, a origem O_R de um raio de pixel R, em WC, é a própria posição da câmera, ou seja,

$$O_R = C, \tag{5.17}$$

e a direção $\mathbf{D}_R$ do raio, como ilustrado na Figura 5.7(a), é dada pelo versor

$$\mathbf{D}_R = \frac{\mathbf{p}}{\|\mathbf{p}\|}, \tag{5.18}$$

em que

$$\mathbf{p} = x_p\mathbf{u} + y_p\mathbf{v} - F\mathbf{n} \tag{5.19}$$

é o vetor da posição da câmera ao ponto P.

Figura 5.7: Raios de pixel para as projeções (a) perspectiva e (b) paralela.

Para a projeção paralela, a direção $\mathbf{D}_R$ de um raio de pixel R é:

$$\mathbf{D}_R = \mathbf{DOP} = -\mathbf{n}. \tag{5.20}$$

A origem O_R do raio, em WC, é, conforme a Figura 5.7(b):

$$O_R = C + x_p\mathbf{u} + y_p\mathbf{v}. \tag{5.21}$$

5.5 Mapeamento Janela/Imagem

Seja um ponto no interior do volume de vista cuja projeção na janela de vista tem coordenadas $(x_p, y_p, -F)$ em relação ao VRC. Para levar a projeção do ponto em um pixel de uma imagem com $m \times n$ pixeis, como ilustrado na Figura 1.1, vamos ignorar a profundidade $-F$ e mapear as coordenadas $(x_p, y_p) \in \mathbb{E}^2$ para o sistema de coordenadas de imagem. As coordenadas (i, j) correspondentes a (x_p, y_p) podem ser obtidas diretamente por interpolação linear. Da Figura 5.8, tem-se

$$\begin{aligned} \frac{i}{m} &= \frac{x_p + \frac{W}{2}}{W} \quad \Leftrightarrow i = \left\lfloor \frac{m}{W}\, x_p + \frac{m}{2} \right\rfloor, \\ \frac{n-j}{n} &= \frac{y_p + \frac{H}{2}}{H} \quad \Leftrightarrow j = \left\lfloor \frac{n}{2} - \frac{n}{H}\, y_p \right\rfloor. \end{aligned} \tag{5.22}$$

O mapeamento inverso (imagem/janela) é:

$$\begin{aligned} x_p &= \frac{W}{m}\, i - \frac{W}{2}, \\ y_p &= \frac{H}{2} - \frac{H}{n}\, j. \end{aligned} \tag{5.23}$$

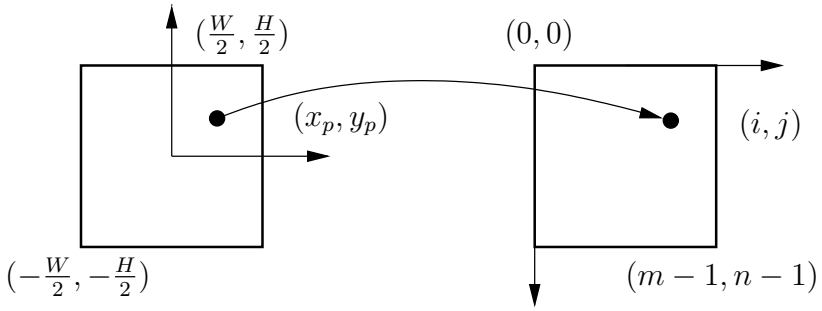

Figura 5.8: Mapeamento janela/imagem.

Observe, nas Equações (5.22), que pontos na borda esquerda ($x_p = -W/2$) da janela são mapeados em pixeis na borda esquerda ($i = 0$) da imagem, como era de se esperar. O mesmo vale para pontos na borda superior ($y_p = H/2$) da janela, os quais são mapeados em pixeis na borda superior ($j = 0$) da imagem. Contudo, para pontos na borda direita ($x_p = W/2$) e inferior ($y_p = -H/2$) da janela, tem-se pixeis com $i = m$ e $j = n$, respectivamente, os quais estão *fora* da imagem. Além disso, como as coordenadas (i, j) são números inteiros resultantes do truncamento de números reais, pontos *distintos* dentro de um mesmo pixel da janela de projeção (exceto as bordas direita e inferior do pixel) mapeiam para o *mesmo* pixel da imagem. (Lembre que o termo *pixel* é usado para denominar cada um dos $m \times n$ pontos de uma imagem, como ilustrado na Figura 1.1, mas também as $m \times n$ *áreas* retangulares resultantes da divisão da janela de projeção para o traçado de raios, veja, por exemplo, a Figura 1.7.) Mais especificamente, todo ponto $(W(i + \delta_x)/m - W/2, H/2 - H(j + \delta_y)/n)$ da janela de projeção, em que $0 \leq \delta_x, \delta_y < 1$, é mapeado para o mesmo pixel (i, j) da imagem, de acordo com as Equações (5.22) (verifique).

No mapeamento inverso, se recuperarmos a profundidade $-F$ descartada na projeção, temos que cada pixel distinto (i, j) da imagem mapeia em um ponto distinto na janela de vista cujas coordenadas VRC são $(x_p, y_p, -F)$, com x_p e y_p dados pelas Equações (5.23). Como mostrado na Figura 5.9, este ponto é o canto superior esquerdo do pixel (i, j) da janela de projeção correspondente ao pixel (i, j) da imagem. Voltando ao algoritmo de traçado de raios introduzido no Capítulo 1, vamos lembrar que o passo 1 traça um raio do observador até o *centro* do pixel $(i.j)$. Mas, se o mapeamento inverso fornece as coordenadas VRC do canto superior esquerdo do pixel (i, j), como determinamos as coordenadas do centro desse pixel? A resposta é: usamos, nas Equações (5.23), $i+0.5$ e $j+0.5$. Note que $(i+0.5, j+0.5)$ *não* são coordenadas válidas de um pixel da imagem, mas são mapeadas para um ponto que é válido, o centro do pixel (i, j) da janela de projeção, de acordo com as Equações (5.23). De fato, todo $(i + \delta_x, j + \delta_y)$, em que $0 \leq \delta_x, \delta_y < 1$, mapeia para pontos distintos do mesmo pixel (i, j) da janela de vista (mais detalhes no Capítulo 6).

Figura 5.9: Mapeamento imagem/janela.

Na OpenGL, as coordenadas mapeadas para imagem não são de pontos projetados na face frontal do volume de vista, mas na face frontal do NDC, isto é, após a divisão de perspectiva. Tanto esta operação quanto a projeção e o mapeamento janela/imagem são efetuados internamente pela OpenGL. A região retangular da tela na qual a imagem será exibida, chamada de *viewport*, é definida através da função `glViewport(`X_v`,`Y_v`,`W_v`,`H_v`)`, em que (X_v, Y_v) são as coordenadas do canto inferior esquerdo (e não do canto superior esquerdo, como na Figura 1.1) e W_v e H_v são a largura e altura da imagem, respectivamente, em pixeis. Assim, um ponto em coordenadas NDC (x_n, y_n, z_n) é mapeado para a *viewport* nas coordenadas (confira)

$$(X, Y) = \left(\left\lfloor \frac{W_v}{2} (x_n + 1) \right\rfloor + X_v, \left\lfloor \frac{H_v}{2} (y_n + 1) \right\rfloor + Y_v \right). \tag{5.24}$$

Além disso, o algoritmo de renderização da OpenGL necessita, para remoção de superfícies escondidas, do valor da profundidade Z de cada vértice mapeado na imagem, também determinada internamente. A função `glDepthRange(`n`,`f`)` pode ser usada para definir os limites de Z (os valores *default* são $n = 0$ e $f = 1$) correspondentes a $z_n = -1$ e $z_n = 1$, respectivamente. Por interpolação linear obtém-se:

$$Z = \frac{f - n}{2} z_n + \frac{f + n}{2}. \tag{5.25}$$

5.6 Operações com a Câmera

Há várias operações com a câmera, como as rotações a seguir.

- Elevação: rotação da posição da câmera em torno de um eixo na direção de versor **u** do VRC e que passa no ponto focal.
- Azimute: rotação da posição da câmera em torno de um eixo na direção de versor **v** do VRC e que passa no ponto focal.
- Rolagem: rotação do **VUP** em torno do eixo z_c do VRC.
- Guinada: rotação do ponto focal em torno do eixo x_c do VRC.
- Arfagem: rotação do ponto focal em torno do eixo y_c do VRC.

Outra operação comum é o *zoom*, que é uma alteração das dimensões da janela de vista por um fator de escala $1/s > 0$. Para $s > 1$, tem-se um *zoom in* (diminuição das dimensões da janela de vista). Para $0 < s < 1$, tem-se um *zoom out* (aumento das dimensões da janela de vista).

5.7 Exercícios

5.1 Seja uma câmera na posição $(3, 4, 0)$ e ponto focal na origem. Dado o versor $\mathbf{v}(0, 1, 0)$ do VRC, determine a origem e direção do raio partindo do observador em direção ao ponto de coordenadas VRC $(1, 1, 0)$, considerando projeção paralela ortográfica e projeção perspectiva.

5.2 Seja uma câmera situada em $(7, 0, 0)$, com ponto focal $(1, 1, 1)$, $\mathbf{VUP}(1, 0, 0)$, ângulo de vista vertical 90^o, razão de aspecto 3/4 e projeção paralela ortográfica.

(a) Após um *zoom in* de $3\times$ (três vezes), quais as coordenadas globais do canto superior direito da janela de vista?

(b) Quais as coordenadas do ponto de coordenadas VRC $(1, 1, 0)$ em uma imagem 1024×768?

(c) Qual a matriz de vista e de projeção da câmera?

(d) Quais são as coordenadas dos versores do VRC após uma elevação de 45^o?

5.3 Quais os parâmetros de uma câmera cujo volume de vista é exatamente o cubo NDC da OpenGL?