Um Resumo das Transformações Geométricas para Visualização em 3D

LUIZ FERNANDO MARTHA [1]
MARCELO GATTASS [2]

Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
ICAD – Laboratório de CAD Inteligente
[1]Departamento de Engenharia Civil
[2]Departamento de Informática
Rua Marquês de São Vicente 225, Gávea
22453-900 Rio de Janeiro, RJ, Brasil
lfm@icad.puc-rio.br

Abstract. The paper describes the derivation and summarizes the geometric transformations which are necessary for displaying a 3D environment on a screen. These transformations are based on the usual set of visualization parameters of the camera model and on the view volume parameters. The main motivation for this article is a didactic derivation of the transformation matrices in a form that is not usually found in text books.

Introdução

O objetivo deste trabalho é mostrar como os parâmetros de visualização do modelo de câmera são utilizados para obter uma imagem projetada e com efeito de perspectiva de um ambiente tridimensional. Esta apresentação tem certamente um caráter didático e resume conhecimentos já estabelecidos na comunidade de Computação Gráfica. A motivação maior para escrever este artigo foi deduzir e resumir as matrizes de transformação geométrica que são utilizadas para visualização em 3D em uma forma didática que não é usualmente encontrada nos livros textos.

Nesta apresentação, as matrizes de transformação 4x4 em coordenadas homogêneas são tais que pré-multiplicam um ponto do modelo a ser transformado (visualizado). Desta forma, a concatenação de uma matriz de transformação à matriz que acumula as transformações sempre se dá na forma de pré-multiplicação.

Parâmetros de visualização em 3D

A abstração utilizada para a especificação dos parâmetros de visualização em 3D segue quase que integralmente um dos modelos definidos pelo *OpenGL* [Neider (1993)], e aqui é definido como **modelo de câmera** (Figura 1). Neste modelo é definido um **frustum de visão** que é uma região na forma de um tronco de pirâmide que limita a porção do espaço tridimensional que é vista através de uma área retângular associada a uma janela na tela de um computador.

Os parâmetros necessários para posicionar a câmera no espaço de modelagem são:

- posição da câmera (olho): $\{eye_x, eye_y, eye_z\}$
- posição do ponto de referência (um ponto no espaço de modelagem para onde a câmera mira): $\{ref_x, ref_y, ref_z\}$
- vetor de orientação vertical da câmera (*view up-vector* – ***vup***): $\{vup_x, vup_y, vup_z\}$

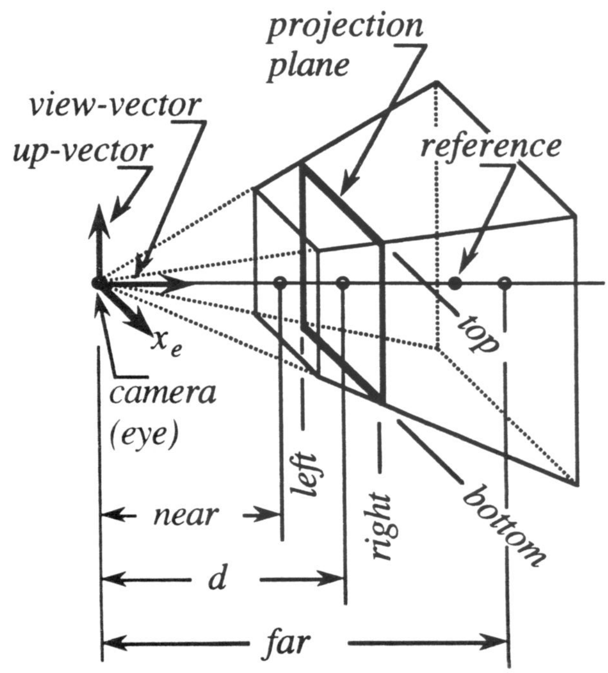

Figura 1: Modelo de câmera e frustum de visão.

O vetor orientado da posição do olho para o ponto de referência é definido como **vetor de visão** (***view***). Os nove parâmetros mostrados acima definem o **sistema de coordenadas do olho**, que é um sistema de coordenadas que tem o ponto do olho como origem, eixo z_e orientado no sentido oposto ao vetor de visão, eixo y_e localizado no plano formado pelos vetores ***view*** e ***vup*** (voltado para o mesmo sentido de ***vup***). Desta forma, os vetores unitários que definem o sistema de coordenadas do olho podem ser definidos como:

$\mathbf{view} = \{ ref_x, ref_y, ref_z \} - \{ eye_x, eye_y, eye_z \}$

$z_e = -\mathbf{view} \,/\, \|\mathbf{view}\|$

$x_e = (\mathbf{vup} \times z_e) \,/\, \|\mathbf{vup} \times z_e\|$

$y_e = z_e \times x_e$

O frustum de visão (Figura 1) pode ser definido de diversas maneiras, e neste trabalho ele é definido pelos limites da janela de visão (*left*, *right*, *bottom*, *top*) em um plano de projeção no sistema de coordenadas do olho e pelas distâncias dos planos de cerceamento anterior (*near*) e cerceamento posteiror (*far*) ao olho, no sentido do vetor de visão. O **plano de projeção** corresponde à janela na tela do computador e é definido pela sua distância (d) ao olho, também no sentido do vetor de visão. Os planos de cerceamento anterior e posterior e o plano de projeção são perpendiculares ao eixo z_e e situam-se do seu lado negativo. Deve-se observar que, enquanto os parâmetros da janela de visão são abscissas com valores reais negativos ou positivos, os parâmetros *far*, *near* e d são por definição distâncias positivas.

É comum associar o plano de cerceamento anterior ao plano de projeção, tal como é feito no *OpenGL* [Neider (1993)].

Também pode-se observar que existe uma redundância de informação ao se especificar a orientação vertical da câmera através de três parâmetros: as três componentes do vetor ***vup***. A função *LookAt* do *GL* [IBM (1990)], por exemplo, especifica a orientação vertical da câmera através de um único parâmetro: o ângulo de giro *twist*. Entretanto a manutenção explícita das três componentes do vetor ***vup*** proporciona uma maneira mais simples de atualizar os parâmetros de visualização durante a manipulação da posição da câmera. Na extensão *glu* do *OpenGL*, a função *LookAt* especifica a orientação vertical da câmera pelas suas três componentes.

Transformação do espaço de modelagem para o sistema da câmera

A primeira transformação geométrica sofrida pelos objetos de um ambiente para serem visualizados é uma mudança de base do sistema de coordenadas de modelagem (do ambiente) para o sistema de coordenadas do olho. Esta mudança de base envolve uma translação da origem do espaço de modelagem para a posição da câmera (olho) e uma rotação do sistema de eixos de modelagem para se igualar ao sistema de eixos do olho.

A matriz que faz a transformação de translação [T] é formada pelas coordenadas da posição do olho no espaço de modelagem:

$$[T] = \begin{bmatrix} 1 & 0 & 0 & -eye_x \\ 0 & 1 & 0 & -eye_y \\ 0 & 0 & 1 & -eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

A matriz que faz a transformação de rotação [R] tem como linhas da sub-matriz 3x3 superior esquerda as componentes dos vetores unitários da base do sistema do olho descritos no sistema de coordenadas de modelagem:

$$[R] = \begin{bmatrix} x_{ex} & x_{ey} & x_{ez} & 0 \\ y_{ex} & y_{ey} & y_{ez} & 0 \\ z_{ex} & z_{ey} & z_{ez} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

A mudança de base do espaço de modelagem para o espaço do olho pode ser grupada em uma única matriz de transformação, definida como **matriz de posicionamento de câmera**: [C] = [R][T].

Perspectiva canônica e espaço da tela

Após a translação [T] e a rotação [R], os objetos estão prontos para serem projetados no plano de projeção, fazendo-se uma projeção cônica (perspectiva) ou ortográfica. A projeção cônica para objetos já transformados para o sistema do olho é denominada de perspectiva canônica. Este tipo de projeção é mostrado na Figura 2 para a coordenada x_e, sendo que para a coordenada y_e o tratamento é análogo.

As coordenadas projetadas de um ponto qualquer $\{x_e, y_e, z_e\}$ são dadas por:

$$x_p = \frac{d}{-z_e} x_e$$

$$y_p = \frac{d}{-z_e} y_e$$

$$z_p = -d$$

As coordenadas $\{x_p, y_p, -d\}$ são as coordenadas de um ponto no espaço de modelagem transformadas e projetadas no plano de projeção da tela. No entanto, não existe nenhuma informação sobre a profundidade do ponto para o interior da tela. Esta profundidade é uma informação importante pois possibilita a correta identificação dos objetos que estão mais próximos do plano de cerceamento anterior e que, por isso, são os objetos visíveis. É, então, criado um espaço de coordenadas $\{x_s, y_s, z_s\}$, chamado de **sistema de coordenadas da tela**, tal que

$$x_s = x_p$$
$$y_s = y_p$$
$$z_s = profundidade$$

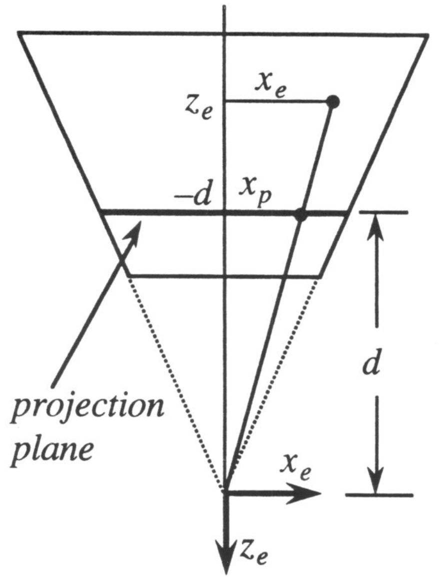

Figura 2: Perspectiva Canônica

Na verdade, para desenhar na tela um ponto $\{x_s, y_s, z_s\}$, precisa-se apenas usar as coordenadas x_s e y_s e ignorar a coordenada z_s. O desenho é uma projeção ortográfica do objeto transformado para o sistema de coordenadas da tela. Isto é, a transformação do objeto para o sistema de coordenadas da tela o distorce de tal maneira que o resultado da projeção ortográfica no plano $x_s y_s$ é igual à transformação de projeção cônica sobre o objeto.

A transformação do objeto para o espaço da tela é conveniente pois faz com que o processo de remoção de linhas e superfícies ocultas de uma imagem seja feito com base em retas perpendiculares ao plano de projeção (projeção ortográfica). Isto é, para saber se um ponto de uma linha ou superfície é obscurecido por outro basta comparar a profundidade z_s dos dois pontos: o que tiver a menor profundidade é o ponto que aparecerá na tela.

Entretanto, para que o resultado da transformação do objeto para o sistema de coordenadas da tela seja útil para a eliminação de linhas e superfícies escondidas, é necessário que se calcule a profundidade de uma linha, não somente para os seus pontos extremos, mas também para qualquer ponto intermediário. Para que a interpolação de um ponto intermediário no espaço da tela seja simples, é preciso que linas retas e planos no sistema de coordenadas do olho sejam transformados para linhas retas e planos no sistema de coordenadas da tela. Pode-se demonstrar [Newman & Sproull (1979)] que a relação que deve existir entre z_e e z_s para que estas condições sejam atendidas é $z_s = \alpha + \beta / z_e$, onde α e β são quaisquer números reais.

Existem infinitos valores de α e β que satifazem a esta imposição de planaridade. Isto é, dados dois valores quaisquer para α e β, tem-se que qualquer reta ou plano no sistema de coordenadas do olho vai se transformar em um plano ou reta no sistema de coordenadas da tela. De fato, o coeficiente α pode ser considerado uma translação de corpo rígido e β um coeficiente influenciando a quantidade de distorção por perspectiva.

Os valores de α e β podem ser escolhidos de tal forma que todos os valores z_e que estão dentro do frustum de visão no sistema de coordenadas do olho mapeiem para valores convenientes de z_s no sistema de coordenadas da tela. Uma solução proposta neste trabalho é manter os valores da coordenada z_e para os planos frontal e posterior de cerceamento no espaço da tela. Isto é, associa-se uma coordenada $z_s = -near$ para pontos com $z_e = -near$ e associa-se $z_s = -far$ para pontos com $z_e = -far$. O resultado disso é:

$$\alpha = -(far + near)$$
$$\beta = -(far \, . \, near)$$

Resumindo, as coordenadas de um ponto no sistema de coordenadas do olho são transformadas para o sistema de coordenadas da tela da seguinte maneira:

$$x_s = \frac{d}{-z_e} x_e$$
$$y_s = \frac{d}{-z_e} y_e$$
$$z_s = -(far + near) - \frac{far \, . \, near}{z_e}$$

Esta transformação pode ser expressa por uma matriz, denominada de **matriz de perspectiva canônica** [P], tal como mostrada abaixo, onde os símbolos (n, f) foram utilizado-se para representar ($near, far$). Nota-se que somente após a divisão pela quarta coordenada homogênea a transformação de um ponto por esta matriz resulta nas expressões acima.

$$[P] = \begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & (f+n) & (f\,n) \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Normalização de coordenadas

Para complementar as transformações geométricas, em geral, é feita uma **normalização** de coordenadas. Esta normalização vai ser útil para a implementação de algoritmos de cerceamento ou remoção de linhas e superfícies ocultas. A exemplo do que é feito no *OpenGL* para esta normalização, a menor coordenada $z_s = -1$ corresponde ao plano $z_e = -near$ e a maior coordenada $z_s = +1$ corresponde ao plano $z_e = -far$. Nota-se que é feito um espelhamento em relação ao plano $x_s y_s$. Isto é dado para que pontos no plano anterior de cerceamento tenham a menor profundidade z_s, e pontos no plano posterior de cerceamento tenham a maior profundidade z_s.

Para projeções ortográficas, a **matriz de normalização de coordenadas** [N] pode ser deduzida a partir de uma transformação de translação do sistema de eixos para o centro do volume de visão, seguida de uma transformação de escala nas três direções, e finalmente de uma transformação de espelhamento em relação $x_s y_s$. Isto resulta na concatenação das duas matrizes mostradas abaixo, onde adotou-se os símbolos (l, r, b, t, n, f) para representar os limites do volume de visão (*left*, *right*, *bottom*, *top*, *near*, *far*):

$$[N] = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & -S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -\Delta_x \\ 0 & 1 & 0 & -\Delta_y \\ 0 & 0 & 1 & -\Delta_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{onde} \begin{cases} S_x = \dfrac{2}{r-l} \\ S_y = \dfrac{2}{t-b} \\ S_z = \dfrac{2}{f-n} \end{cases} \quad \text{e} \begin{cases} \Delta_x = \dfrac{r+l}{2} \\ \Delta_y = \dfrac{t+b}{2} \\ \Delta_z = -\dfrac{f+n}{2} \end{cases}$$

A matriz de normalização para projeções ortográficas que resulta é:

$$[N] = \begin{bmatrix} \dfrac{2}{r-l} & 0 & 0 & -\dfrac{r+l}{r-l} \\ 0 & \dfrac{2}{t-b} & 0 & -\dfrac{t+b}{t-b} \\ 0 & 0 & \dfrac{-2}{f-n} & -\dfrac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Para projeções cônicas, pode-se obter uma única matriz de transformação que engloba a perspectiva canônica [P] e a normalização de coordenadas [N]. Esta é a chamada **matriz de frustum** [F] = [N][P].

A concatenação de [P] e [N] para a obtenção de [F] pode ser entendida da seguinte maneira:

- A primeira transformação [P] a ser aplicada distorce o frustum de visão de um tronco de pirâmide para um prisma, onde as coordenadas máximas e mínimas z_e nos planos de cerceamento anterior e posterior são preservadas. As outras coordenadas máximas e mínimas deste prisma correspondem à janela de visão (*left*, *right*, *bottom*, *top*) definida no plano de projeção.
- A transformação [P] distorceu o frustum de visão para um prisma e fez o problema recair em uma normalização de projeções ortográficas. Portanto, basta aplicar a transformação [N] para normalizar as coordenadas.

A matriz [F] resultante é mostrada abaixo. No *OpenGL* esta matriz é apresentada com o parâmetro *near* (n) no lugar do parâmetro d pois é assumido que o plano de projeção é o plano de cerceamento anterior.

$$[F] = \begin{bmatrix} \dfrac{2d}{r-l} & 0 & \dfrac{r+l}{r-l} & 0 \\ 0 & \dfrac{2d}{t-b} & \dfrac{t+b}{t-b} & 0 \\ 0 & 0 & -\dfrac{f+n}{f-n} & \dfrac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Conclusão

Pode-se resumir a seqüência de operações geométricas que um ponto no espaço de modelagem sofre para ser transformado para o espaço normalizado da tela. Isto é feito através da seguinte concatenação de matrizes:

$\{p_s\} = [N][P][R][T]\ \{p\}$

$\{p_s\} = [F][C]\ \{p\}$

A concatenação com a matriz [P] não é feita no caso de projeções ortográficas.

Referências

W. Newman, R. Sproull, *Principles of Interactive Computer Graphics*, 2nd ed., McGraw-Hill, 1979.

J. Neider, T. Davis, M. Woo, *OpenGL Programming Guide – The Official Guide to Learning OpenGL*, Release 1, Addison Wesley, 1993.

IBM Manual AIX Version 3 for RISC System/6000, *Graphics Programming Concepts*, 1990.