Visualização e Transformações Perspectiva

Foley & van Dam - Cap. 6
Notas de aula do Prof. Mount: aulas 9 e 10

3D no OpenGL

- Para gerar imagens de um objeto 3D, é necessário compreender transformações perspectiva
- No caso do OpenGL, essa transformação é apenas uma parte do processo de visualização
- Vamos considerar que os objetos são representados em um frame de coordenadas 3D, que vamos chamar de coordenadas do mundo

Processo

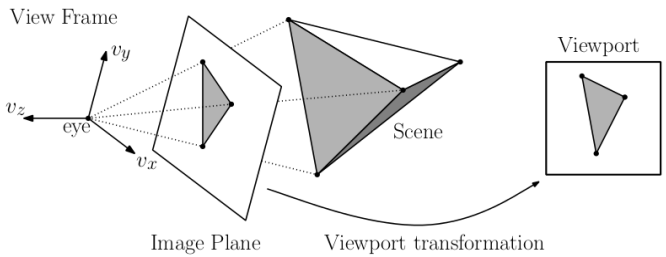


Fig. 29: OpenGL Viewing Process.

Transformações do Modelview

- Mapeia um objeto (seus vértices) de sua representação em coordenadas do mundo para outra representação centrada no observador.
  - coordenadas do observador, ou
  - coordenadas da câmera,
  - ou do olho.

Projection Matrix

- Pontos em coordenadas 3D da câmera são mapeados para o plano de imagem.
- Esse processo exige:
  - transformação projetiva
  - clipping
  - normalização perspectiva
- A saída é gerada em coordenadas normalizadas do dispositivo.
  - veja: gluOrtho2D, glOrtho, glFrustum, e gluPerspective

Mapear para o Viewport

- Pontos nas coordenadas normalizadas do dispositivo são convertidas para o viewport.
  - coordenadas do viewport, ou
  - coordenadas da janela

  - veja o comando: glViewport

Usando coordenadas centradas no observador

- Projeção perspectiva: mais fácil de fazer quando o centro de projeção é a origem e o plano de imagem é ortogonal a um eixo, e.g., o eixo z.
- No entanto, para construir o desenho (cena 3D) nós usamos o sistema de coordenadas do mundo (arbitrário, escolhido pelo usuário)

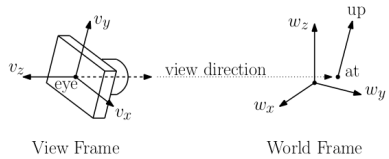
Como fazer a
transformação perspectiva
usando as coordenadas
do mundo?

gluLookAt()

- transforma a cena para coordenadas centradas no observador
- deve ser feita portanto no Modelview

Exemplo típico

```
def myDisplay():
    glClear( GL_COLOR_BUFFER_BIT |
             GL_DEPTH_BUFFER_BIT )
    glLoadIdentity()
    gluLookAt (eyeX, eyeY, eyeZ,  atX, atY, atZ,  upX, upY, upZ)
    # desenhe o resto da cena
    glutSwapBuffers()
```



Coordenadas da camera

- eixo x aponta para a direita
- eixo y aponta para cima
- eixo z aponta para traz (por que?)
  - mantém o sistema "mão direita"

exercício:
construa o frame de coordenadas
da câmera a partir dos parâmetros
do gluLookAt()

Projeções

- Grupos básicos:
  - Paralela: todas as linhas de projeção são paralelas
    - geometria: obdece transformações afims
  - Perspectiva: as linhas de projeção convergem em um ponto
    - geometria projetiva
    - linhas são mapeadas para linhas

Geometria Projetiva

Princípios

- Geometria projetiva
  - estuda como projetar pontos de um espaço d dimensional para um hiper-plano de d-1 dimensões (plano de projeção) através de um ponto (fora do plano) chamado centro de projeção.
- Propriedades:
  - linhas são mapeadas para linhas
  - não preservam combinações afins (exemplo de ponto central).

Geometria projetiva

- Desenvolvida no século 17 por matemáticos interessados no fenômeno de perspectiva.
- Geometria euclidiana: duas linhas se intersectam em apenas um ponto, se não paralelas.
- Extensão: além de pontos regulares, considere um conjunto de pontos ideais (ou pontos no infinito).
  - é possível definir então que duas retas distintas sempre tem um ponto em comum. Quando paralelas, o ponto em comum é um ponto ideal.

Geometria projetiva

Por que não dois pontos?

- quantas vezes você quer que 2 retas se cruzem?

- imagine assim que o plano projetivo se dobre de forma que dois pontos ideais em lados opostos se encontrem, ou melhor, sejam o mesmo ponto.

Pontos ideais

- Com a introdução de pontos ideais, cada reta se comporta como uma curva fechada (semelhante a um círculo de raio infinito).
- Um ponto ideal P tem uma posição (no sentido de espaço afim), e ele pode ser definido por uma direção v (apontando para o ponto).
  - a família de retas paralelas a v se encontram em P
  - no plano, a união de todos os pontos ideais formam uma linha no infinito. Em 3D, formam um plano no infinito.
  - note que cada linha intersecta a linha no infinito uma única vez.

Plano projetivo

- O plano projetivo é formado pelo plano afim mais os pontos ideais e a linha no infinito (pode-se estender o conceito para outras dimensões).
- Bastante elegante, porém coisas estranhas acontecem. Em particular, inversão de orientação quando um objeto orientado passa por infinito.
- Por isso, usamos o plano projetivo apenas quando necessário, e sempre que possível trabalhamos dentro do nosso "familiar" espaço euclidiano.

Coordenadas Homogêneas

- Algumas diferenças do espaço afim:
  - não há vetores livres no espaço projetivo. Eles serão utilizados na construção de objetos, porém não são projetados.
  - Um ponto regular P no plano tem coordenadas não homogêneas (x,y), e pode ser representado por (w.x,w.y,w)T, w != 0 (lembre-se que antes w=1)
    - ou seja: se P = (4,3) no espaço cartesiano, suas coordenas homogêneas podem ser representadas por (4,3,1), (8,6,2), (-12,-9,-3), etc.
    - w = 1 será usado com frequência, apenas lembre-se que no espaço projetivo basta w ser diferente de 0.

Normalização

- Dadas as coordenadas homogêneas de um ponto P = $(x, y, w)^T$, a normalização de P é o vetor de coordenadas $(x/w, y/w, 1)^T$
- Observe que normalização nesse caso não significa criar um vetor normal de módulo 1.
- Para evitar essa confusão, chamaremos esse processo de divisão perspectiva

Pontos ideais

- Considere uma reta passando pela origem com inclinação 2.
- A coordenada homogênea de alguns pontos nessa linha pode ser dada por: (1,2,1), (2,4,1), (3,6,1), (4,8,1), ...
- Que são equivalentes a: (1,2,1), (1,2,1/2), (1,2,1/3), (1,2,1/4), ...
- Pode-se observar que no limite teremos (1,2,0). Assim, quando w=0, temos o ponto (x,y,w) como sendo o ponto no infinito, apontado pelo vetor (x,y)

Transformações projetivas

- Utilizaremos matrizes 4x4 como no espaço 3D afim, com algumas alterações. Em particular, apenas pontos serão transformados, como os extremos de segmentos de linha, ou vértices de um polígono.
- Algumas considerações:
  - Apenas pontos na frente do olho serão considerados
  - Simplificação: usaremos desenhos no plano y-z. Por simetria, tudo que será mostrado nesse plano, vale para o plano x-z

Sistema de coordenadas

- Centro de projeção na origem
- O observador está olhando na direção de -z (sistema da mão direita).
- O eixo x aponta para a direita do observador
- O eixo y aponta para cima
- Plano de projeção distante d da origem, ortogonal a z. Note que d é positivo, enquanto z é negativo.
- Dessa forma um ponto P = (y,z) no plano, será projetado no ponto (yp, zp) = (yp, -d), onde y/-z = yp/d
- Em 3D: (x/(-(z/d)), y/(-(z/d)), -d, 1) T

Note que não há uma matriz 4x4 capaz de fazer a transformação $(x/(-(z/d)), y/(-(z/d)),-d,1)T$.

Por que?

Matriz de transformação

- Pois nos termos x e y, o valor de z não é uma constante, e portanto não pode ser armazenada na matriz.
- Modificando um pouco a expressão, multiplicando os elementos por -(z/d), temos: $(x,y,z,-z/d)T$
  - essa forma é linear em (x,y,z), e portanto pode ser expressa em forma matricial, tal que P' = M . P
  - Obtido as coordenadas de P', aplicamos a divisão perspectiva para obter um ponto no espaço euclidiano.
  - E quando z = 0?
    - divisão por zero. Pontos no plano z=0 são mapeados no infinito.

Perspectiva com Profundidade

- Observe a forma: $(x/(-(z/d)), y/(-(z/d)), -d, 1)^T$
  - as duas últimas componentes são ctes, e portanto não contribuem com nenhuma informação.
  - Após a projeção, toda informação de profundidade é perdida, porém, essa é uma informação necessária para remover as superfícies escondidas. Como então incluir essa informação?
- Faremos uso da transformação perspectiva com profundidade:
  - transforma as coords (x,y) de um ponto para as suas respectivas coords projetivas
  - coordenada z codifica a informação de profundidade

Forma matricial

- Obs: a informação de profundidade será distorcida de uma forma não linear, porém, a ordem relativa ao observador é preservada.
- Assumindo o olho na origem e o plano de projeção em z=-1, considere a matriz:
  - $M = ((1\ 0\ 0\ 0)T \mid (0\ 1\ 0\ 0)T \mid (0\ 0\ \alpha\ -1)T \mid (0\ 0\ \beta\ 0)T)$
- Aplicando M para um ponto P = (x, y, z, 1)T, em coordenadas homogêneas, teremos que
  - $M.P = (x \quad y \quad (\alpha.z+\beta) \quad -z)T$
  - depois da divisão perspectiva, temos que a profundidade é dada por: $z' = (\alpha\ z + \beta)/-z = (-\alpha - \beta/z)$
  - desejamos que a função z' seja monotônica

Volume canônico de observação

- A transformação perspectiva é aplicada a todos os pontos, inclusive aqueles fora do frustum.
- O sistema deve portanto recortar as porções da cena fora do frustum, antes de gerar a imagem final (na verdade antes da divisão perspectiva).
- Para simplificar e tornar o clipping eficiente, o OpenGL se utiliza de um volume canônico de observação, que é sempre o mesmo!
- Clipping no OpenGL é feito em coordenadas homogêneas, antes da divisão perspectiva.

Clipping

- Por simplificação, descreveremos o volume canônico após a divisão perspectiva
  - O volume canônico se torna um paralelogramo definido por: -1 <= x,y,z <= +1
- As coordenadas (x,y) são as coords dos pontos projetados na janela de observação.
- z é usada para representar a profundidade
  - z = -1 pontos mais próximos do observador
  - z = +1 pontos mais distantes.

Matriz Perspectiva

- sejam
  - c = cot (θ/2)
  - a = aspect ratio
  - n | f = distâncias ao plano near | far
- A matriz de transformação perspectiva M é:
  - M=((c/a 0 0 0)T |(0 c 0 0)T |
    (0 0 ((f+n)/(n-f)) -1)T |(0 0 (2fn/(n-f)) 0)T)
  - esses parâmetros foram escolhidos para criar o volume canônico de observação (vco)
- Exercício: verifique a matriz de transformação para os cantos do vco (pontos (1,1,-1,1) e (-1,-1,1,1) por exemplo).

Resumo:

- Geometria projetiva
- Pontos ideais
- Plano projetivo
- Coordenadas homogêneas
- Transformações
- Sistemas de coordenadas
- Perspectiva no OpenGL
  - Volume canônico de transformação
  - Clipping