Transformarea de vizualizare din spatiul 3D

Prof. univ. dr. ing. Florica Moldoveanu

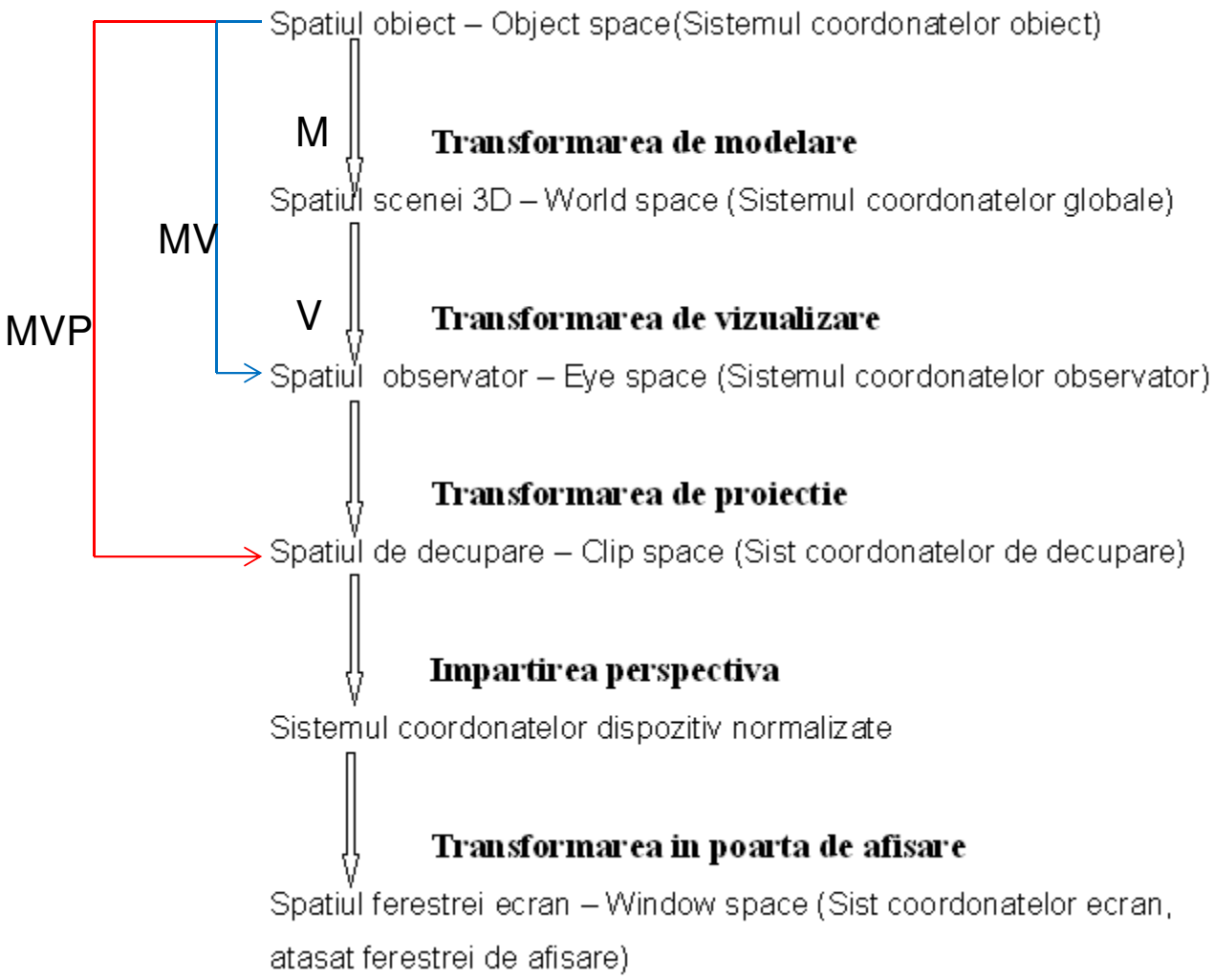
Transformarea de vizualizare din spatiul 3D

- Are ca scop transformarea modelului unei scene 3D intr-o imagine redata pe o suprafata de afisare (ecran).
- Poate fi comparata cu transformarea efectuata de un aparat foto asupra lumii reale pentru a se obtine imaginea dintr-o fotografie:
  - Imaginea obtinuta depinde de:
    - pozitia ochiului(observatorului)
    - directia in care priveste(observatorul)
    - ceea ce vede ochiul prin vizorul aparatului foto (volumul vizual)

- Este o transformare compusa din mai multe transformari, intre care si o proiectie $R^3 \rightarrow R^2$
- Se efectueaza asupra varfurilor obiectelor din scena 3D.
- Include si operatia de decupare (clipping): eliminarea partilor din scena 3D care sunt in afara volumului vizual.

- OpenGL efectueaza automat transformarea de vizualizare asupra varfurilor obiectelor 3D descrise intr-o aplicatie, existand intotdeauna o matrice curenta a transformarii de vizualizare din spatiul 3D.

Transformarea varfurilor in OpenGL



Transformarea de modelare (1)

Spatiul obiect

- In mod uzual, fiecare obiect este definit in propriul sau sistem de coordonate (sistemul coord locale – 3D « dreapta »); exemple: cubul, sfera, scheletul unui personaj animat, s.a.
- Avantaje: usurinta modelarii si reutilizarea modelelor obiectelor 3D

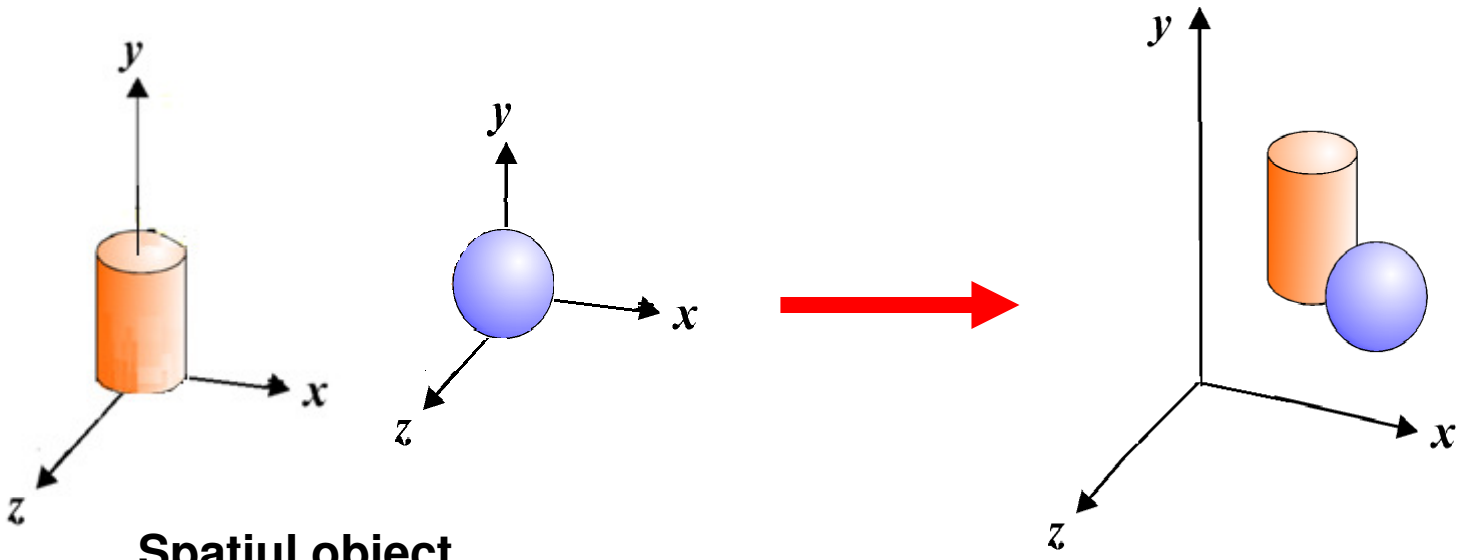
Spatiul scenei 3D (Spatiul lumii reale -World space)

- Spatiul in care este compusa scena 3D
- Raportat la un sistem de referinta global – sist de coord carteziene 3D “dreapta”
- Originea sistemului de coord este stabilita de modelatorul scenei 3D

Transformarea de modelare

- ❖Scop: dimensionarea si poziţionarea obiectelor în scena 3D.
- ❖Este o transformare geometrica compusa.

Transformarea de modelare (2)



Spatiul obiect

Spatiul scenei 3D

Obiecte definite in sisteme de coordonate proprii (sisteme de coordonate locale)

Transformarea de modelare

Obiecte definite fata de acelasi sistem de coordonate (sistemul de coordonate globale)

Translatie, Rotatie, Scalare

Transformarea de vizualizare (1)

Spatiul observator (Eye space)

- O imagine este o vedere obtinuta dintr-un punct al spatiului 3D, similara unei fotografii:
- Depinde de:
  - Pozitia observatorului in spatiul scenei,
  - Directia in care priveste si ceea ce observatorul reuseste sa vada prin vizorul aparatului de fotografiat.
  - Rotatia aparatului fata de axa sa.

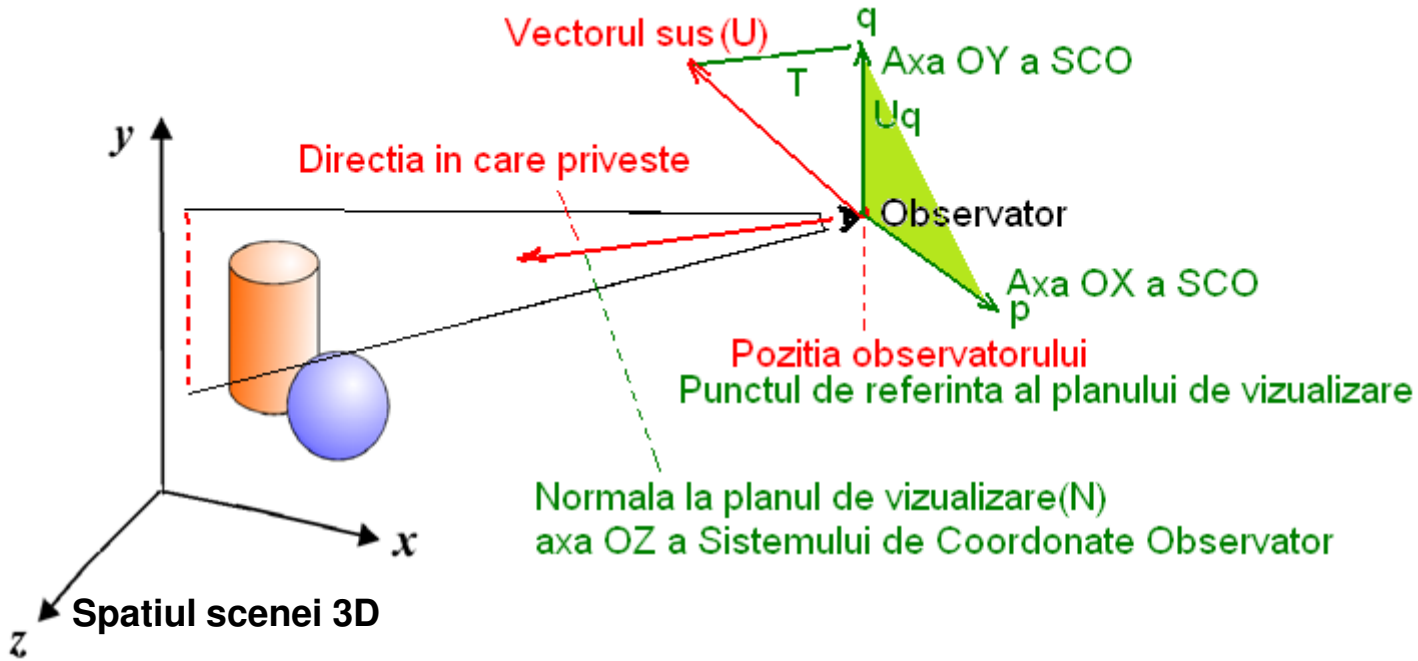
Sistemul de coordonate observator: atasat planului de vizualizare

❖ **Sistem de coordonate 3D stanga, definit prin 3 parametri**:

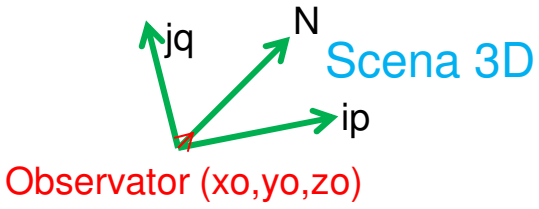
- Originea sa: pozitia observatorului (ochiului) – aflata in planul de vizualizare
- Normala la planul de vizualizare : directia axei OZ a sistemului de coordonate observator- este directia in care priveste observatorul (centru scenei 3D)
- Directia sus a planului de vizualizare, care determina axa OY a sist. de coordonate

Transformarea de vizualizare (2)

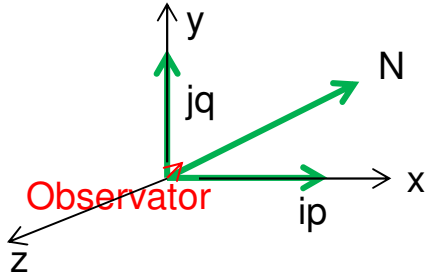
din sistemul coordonate globale → in sistemul de coordonate observator



Spatiul scenei 3D



Sistemul de coordonate observator (SCO)



Sistemul de coordonate observator in OpenGL

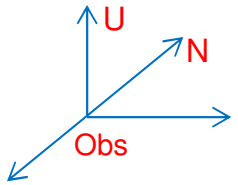
Transformarea de vizualizare (3)

Determinarea sistemului de coordonate observator folosind cei 3 parametri: CURS

In OpenGL, valorile implicite ale parametrilor care definesc sistemul de coordonate observator sunt:

- poziţia observatorului (camera de luat vederi): **originea sistemului de coordonate globale**
- direcţia în care priveşte observatorul: **direcţia negativa a axei OZ**
- direcţia sus a planului de vizualizare: **direcţia pozitivã a axei OY**

Cu aceste valori implicite, transformarea de vizualizare este transformarea « dreapta-stanga »: x' = x, y' = y, z' = -z.



Funcţia *gluLookAt()* permite modificarea valorilor implicite ale parametrilor transformării de vizualizare (care definesc sistemul de coordonate observator):

void **gluLookAt**(GLdouble eyex,GLdouble eyey,GLdouble eyez, GLdouble centrx, GLdouble centry, GLdouble centrz, GLdouble upx,GLdouble *upy,* GLdouble upz);

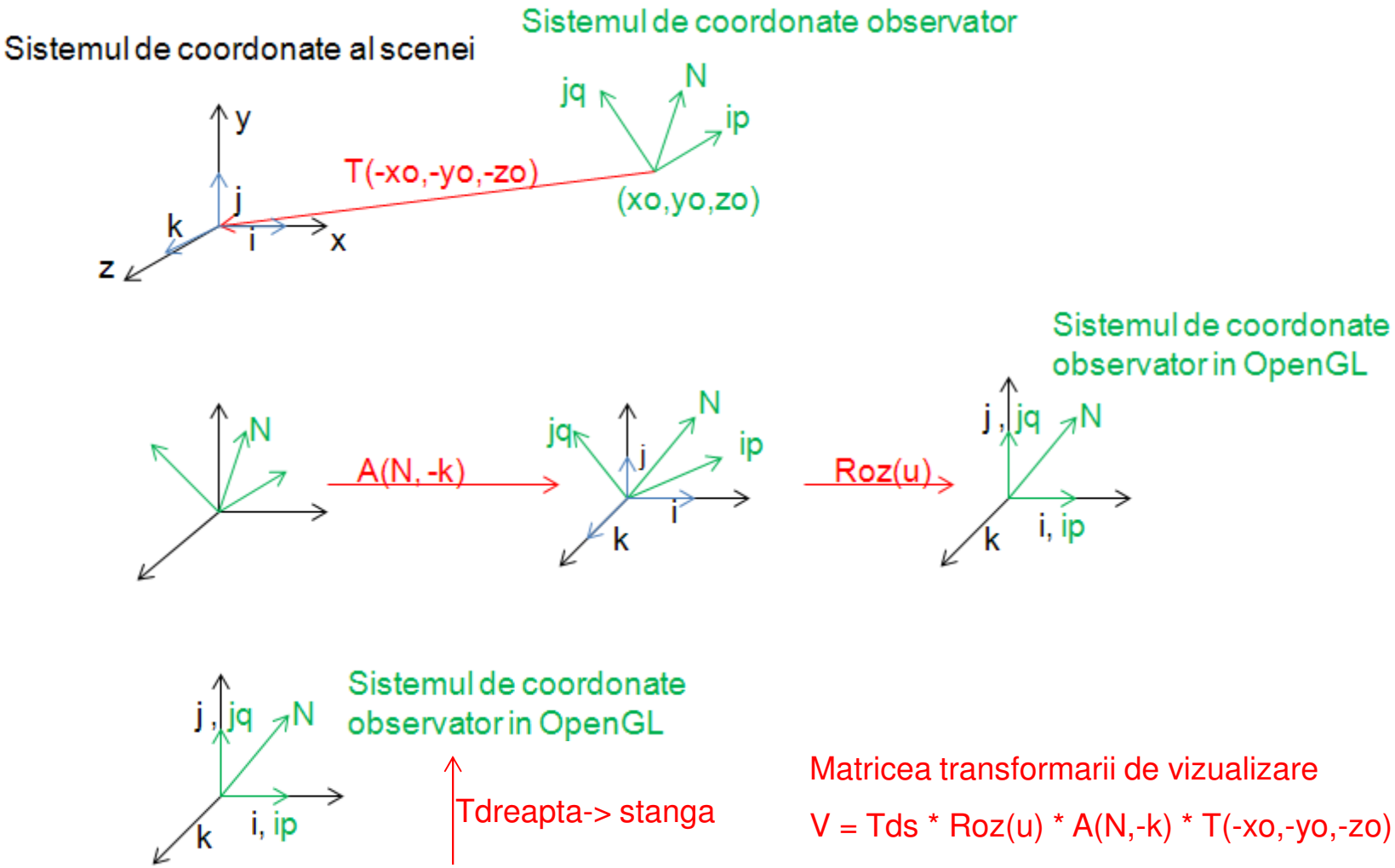
unde:

eyex, eyey, eyez reprezintă **poziţia observatorului**

centerx, centery, centerz reprezintă **direcţia în care se priveşte observatorul (centrul scenei 3D)**

upx, upy, upz reprezintă **direcţia vectorului « sus » al planului de vizualizare**

Transformarea de vizualizare (4)



Matricea transformarii de vizualizare

V = Tds * Roz(u) * A(N,-k) * T(-xo,-yo,-zo)

Transformarea de modelare si vizualizare

din sistemul de coordonate obiect → in sistemul de coordonate observator

- Transformare compusa (model-view) reprezentata printr-o singura matrice in OpenGL

  VM = V•M

  M – matricea de modelare curenta

  V – matricea de vizualizare curenta

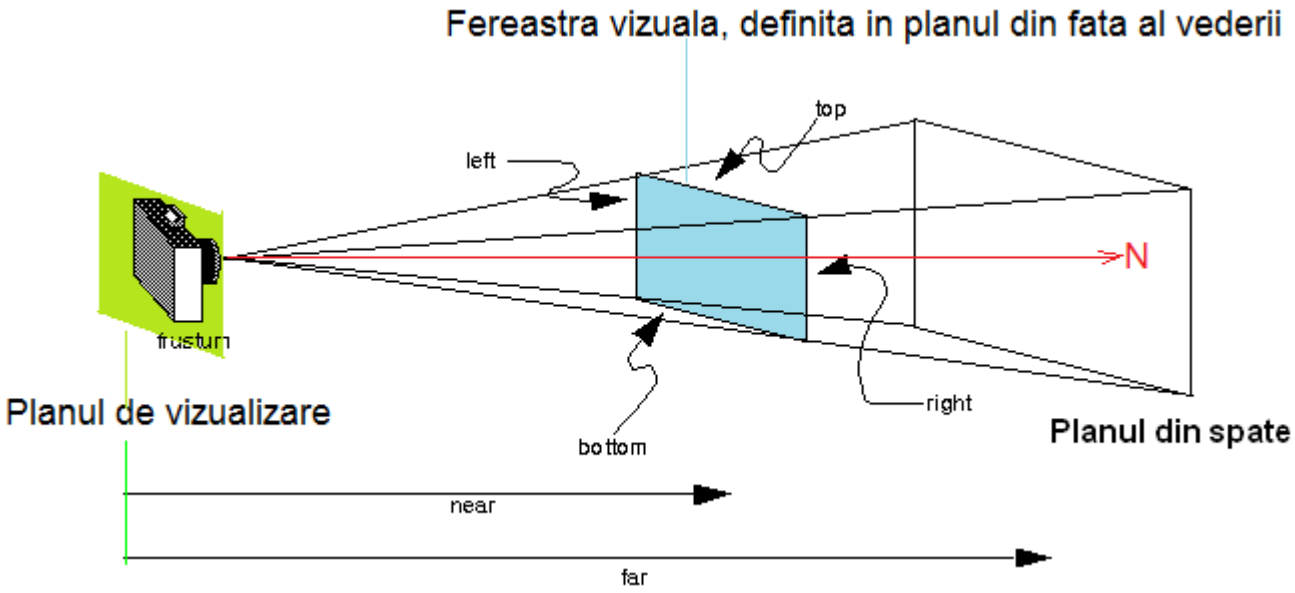
- Inmultirea celor 2 matrici este efectuata automat de OpenGL: exista intotdeauna o matrice VM!

$[xe\ ye\ ze\ we]^T = VM \bullet [xo\ yo\ zo\ wo]^T$

Transformarea de proiectie (1)

- Determinata de **volumul vizual** definit de programator:
  - trunchi de piramida pentru proiectia perspectiva
  - paralelipiped dreptunghic pentru proiectia paralela

Volumul vizual al proiectiei perspectiva: trunchi de piramida delimitat de planul din fata (near) si planul din spate (far) al vederii; near si far – valori pozitive masurate pe axa de profunzime a planului de vizualizare.



Transformarea de proiectie (2)

In OpenGL, volumul vizual al proiectiei perspectiva poate fi specificat prin apelul functiei:

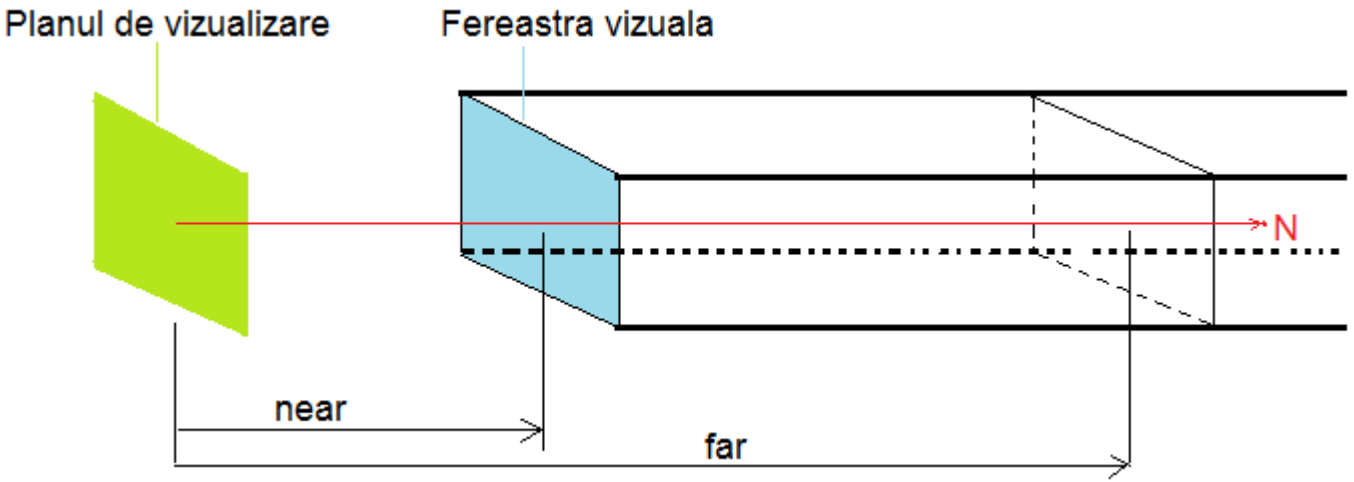
void **glFrustum**(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);

-Funcţia defineşte o proiecţie perspectivă cu centrul de proiecţie în poziţia observatorului .

Volumul vizual al proiectiei ortografice

- Directia de proiectie este directia axei N a planului de vizualizare

- Volumul vizual este un paralelipiped dreptunghic cu laturile paralele cu directia de proiectie, delimitat in adancime de planul din fata (near) si planul din spate (far) al vederii



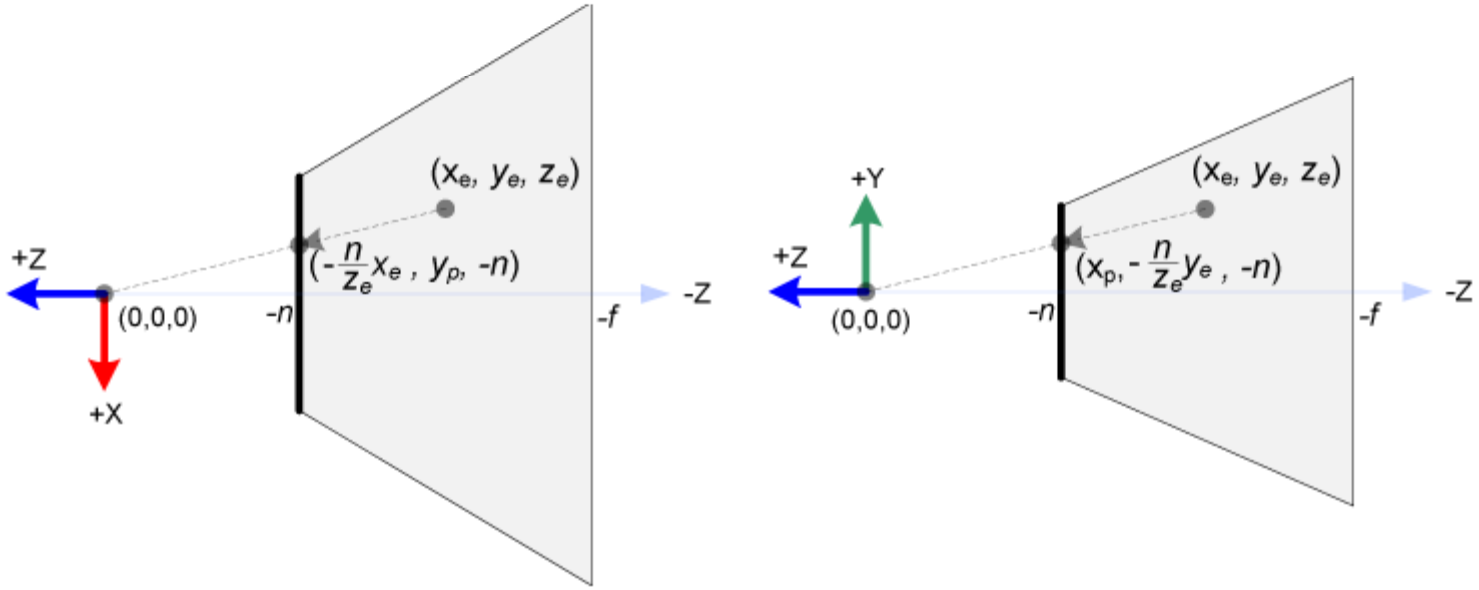
Transformarea de proiectie (3)

In OpenGL, volumul vizual al proiectiei ortografice poate fi specificat prin apelul functiei:

void **glOrtho**(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);

- **Pentru ambele tipuri de proiectii, proiectia se efectueaza in (fereastra din) planul din fata: z=-near**

Proiectia perspectiva a unui punct in coordonate observator, (xe,ye,ze), pe planul din fata al vederii, cu centrul de proiectie in (0,0,0).



xp = (-near/ze)*xe, yp =(-near/ze)*ye – se pot obtine plecand de la formulele proiectiei perspectiva in XOY sau din asemanarea triunghiurilor: xp/xe = -n/ze, yp/ye = -n/ze.

Transformarea de proiectie (4)

Proiectia ortografica a unui punct in coordonate observator, (xe,ye,ze), pe planul din fata al vederii:

xp = xe, yp = ye, zp = -near.

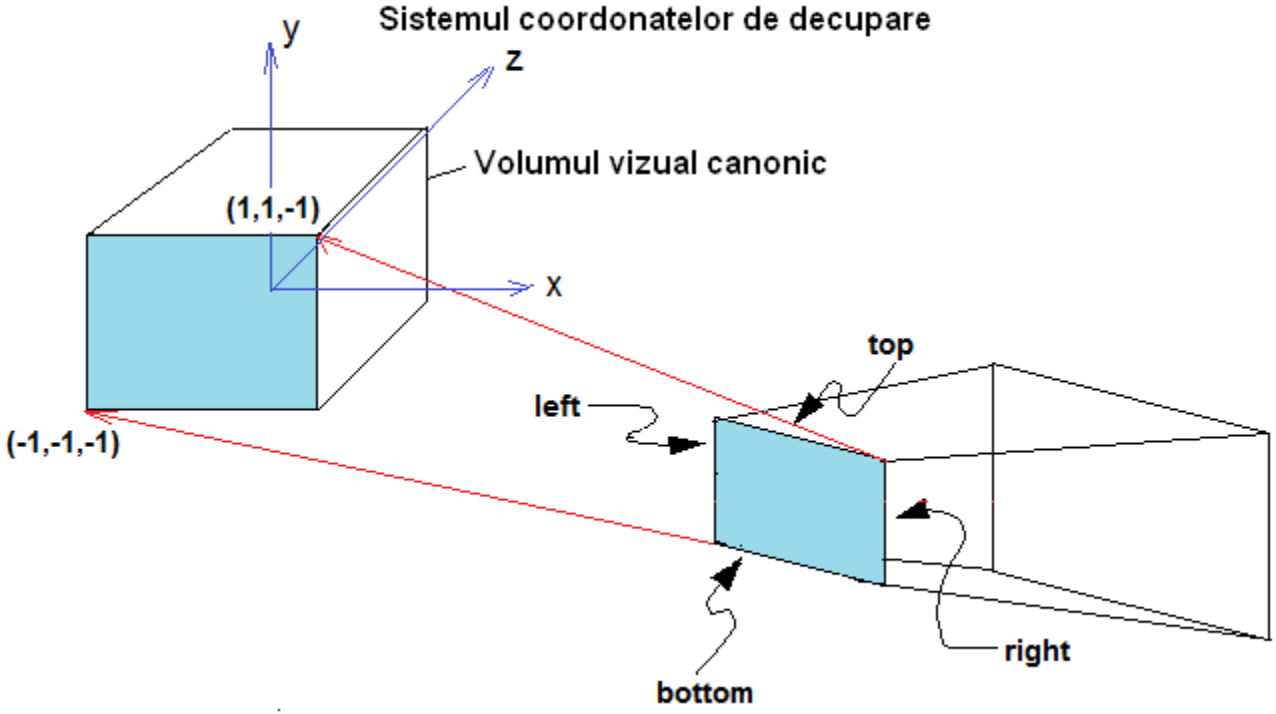
- Transformarea de proiectie este insotita de operatia de decupare a primitivelor grafice la marginile volumului vizual.
- Pentru simplificarea calculelor de decupare, volumul vizual definit de programator este transformat (de OpenGL) in volumul vizual canonic: cub cu latura de 2 unitati, centrat in originea sistemului de coordonate de decupare, cu laturile paralele cu axele.
- Transformarea volumului vizual in volumul vizual canonic se numeste transformarea de normalizare = transformarea de proiectie, in OpenGL.

Dupa transformarea varfurilor prin transformarea de proiectie, primitivele grafice 3D sunt decupate la marginile volumului vizual canonic.

Transformarea de proiectie (5)

- *Sistemul coordonatelor de decupare*: sistem de coordonate 3D stanga

- Prin transformarea de proiectie, colţurile ferestrei 2D din planul de proiecţie, (left, bottom,-near) şi (right, top,-near) sunt mapate pe colţurile stânga-jos şi dreapta-sus ale ferestrei 2D din sistemul coordonatelor de decupare, adică (-1,-1,-1) şi (1,1,-1).



Transformarea de proiectie (6)

Impunand conditia (left,right) → [-1,1] si [bottom,top] → [-1,1], rezulta transformarea punctelor din planul din fata, (xp,yp), in sistemul coordonatelor de decupare (**clip coordinates**):

xd = 2*xp/(right – left) – (right + left)/(right – left)

yd = 2*yp/(top – bottom) – (top + bottom)/(top-bottom)

Inlocuind xp si yp rezulta:

xd = (xe*2*near/(right – left) + ze*(right + left)/(right – left))/(-ze)

yd = (ye*2*near/(top – bottom) + ze*(top + bottom)/(top-bottom))/(-ze)

Pentru transformarea coordonatelor z se impun conditiile: -near → -1, -far → 1;

Rezulta:

zd = (ze*(far+near)/(far-near) – 2*far*near/(far-near)) / (-ze)

Punctul in coordonate omogene: (xc, yc, zc, w)

xd = xc/w; yd = yc/w; zd = zc/w ; w = - ze

Transformarea de proiectie (7)

Matricea de proiectie pentru proiectia perspectiva este:

$$P = \begin{bmatrix} \frac{2 * near}{right - left} & 0 & A & 0 \\ 0 & \frac{2 * near}{top - bottom} & B & 0 \\ 0 & 0 & C & D \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

$$A = \frac{right + left}{right - left} \qquad B = \frac{top + bottom}{top - bottom}$$

$$C = -\frac{far + near}{far - near} \qquad D = -\frac{2 * far * near}{far - near}$$

Pentru proiectia ortografica

xp=xe si yp=ye → xd = 2*xe/(right – left) – (right + left)/(right – left)

yd = 2*ye/(top – bottom) – (top + bottom)/(top-bottom)

Matricea de proiectie pentru proiectia ortografica este:

$$P = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -A \\ 0 & \frac{2}{top - bottom} & 0 & -B \\ 0 & 0 & \frac{-2}{far - near} & C \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Transformarea de proiectie (6)

din coordonate observator → in coordonate de decupare

$$[xc\ yc\ zc\ w]^T = P \bullet [xo\ yo\ zo\ wo]^T$$

Impartirea perspectiva:

xd = xc/w , yd = yc/w , zd= zc/w

(xd, yd, zd): coordonate dispozitiv normalizate

Transformarea de modelare-vizualizare-proiectie

❖ Matricile transformarilor de modelare, vizualizare si proiectie sunt inmultite de OpenGL, rezultand transformarea compusa numita “model-view-projection”, a.i. asupra varfurilor definite in sistemul coordonatelor obiect este efectuata o singura transformare:

PVM = P•V•M (matricea ModelViewProjection)

Transformarea in poarta de afisare

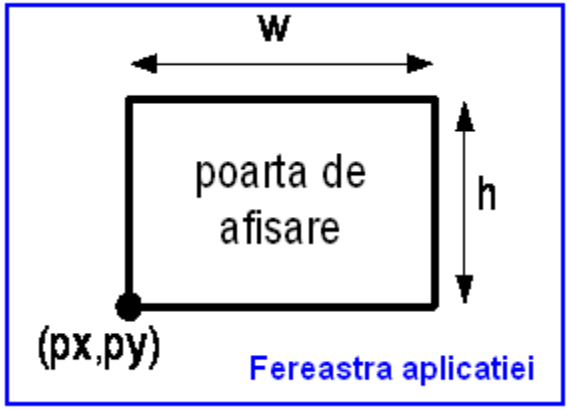
- Transformare fereasta-poarta
- Fereastra este fereastra 2D din sistemul coordonatelor de decupare,

  având colţurile în (-1, -1, -1) - (1, 1, -1)
- Poarta este definita de programator folosind functia:

  void **glViewport**(GLint px, GLint Py, GLsizei width, GLsizei height);

  (*px, py*) reprezintă coordonatele în fereastra ecran a aplicatiei ale colţului stânga jos al porţii de afişare (în pixeli). Valorile implicite : (0,0).

  width, height reprezintă lăţimea, respectiv înălţimea porţii de afişare. Valorile implicite sunt date de lăţimea şi înălţimea ferestrei aplicatiei.



- Transformarea in poarta de afisare este definita astfel:

  xw = ox + (width/2)xd

  yw = oy + (height/2)yd

  zw = ((f-n)/2)zd + (n+f)/2 // conserva coordonata z pentru eliminarea partilor nevizibile

  (ox, oy) reprezintă coordonatele centrului porţii de afişare

  width, height reprezintă lăţimea, respectiv înălţimea porţii de afişare

  n şi f au valorile implicite 0.0 respectiv 1.0, dar pot fi modificate la valori cuprinse în intervalul [0,1] folosind funcţia ***DepthRange()***