Transformarea de vizualizare din spatiul 3D

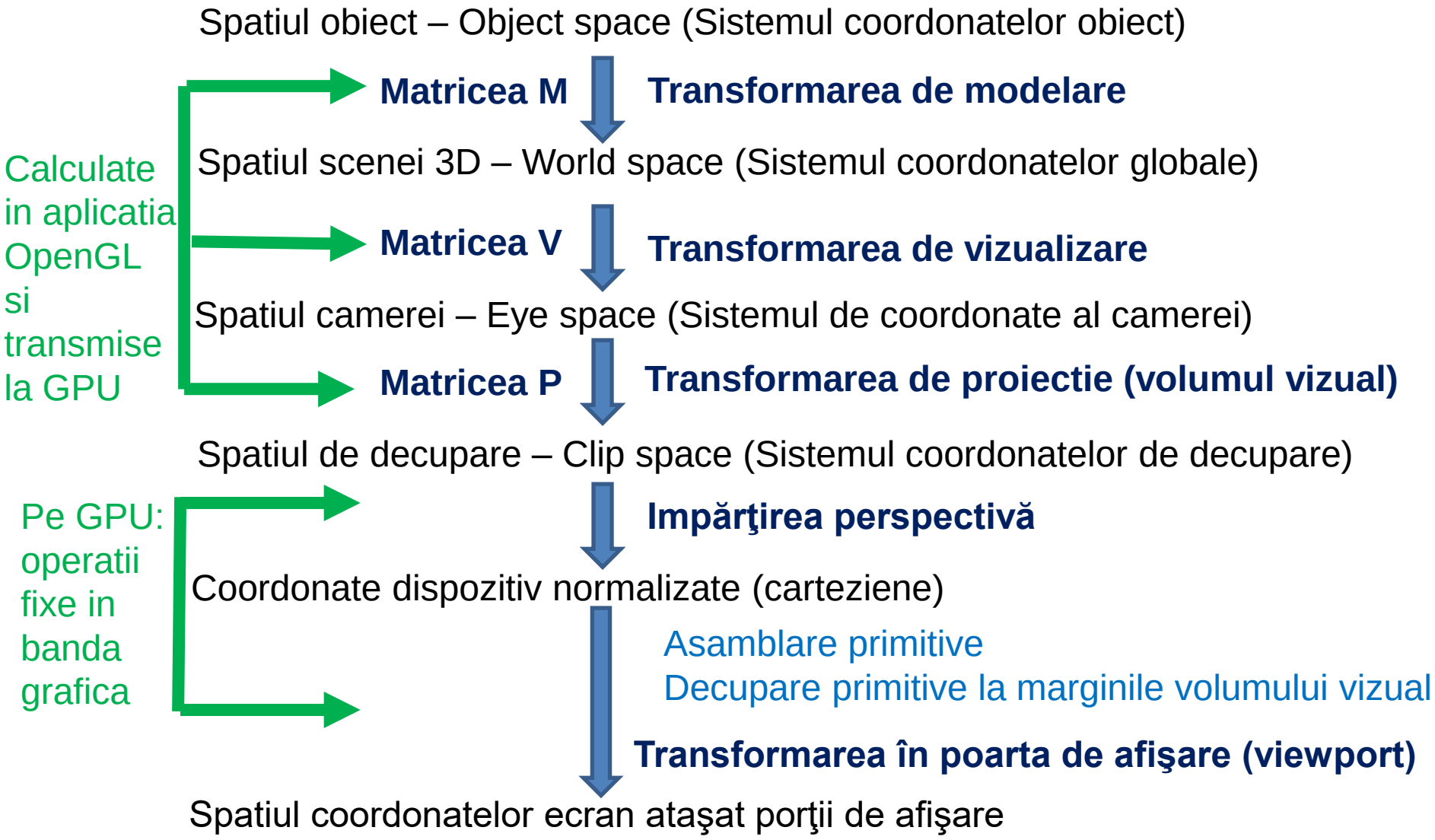
Prof. univ. dr. ing. Florica Moldoveanu

Curs *Elemente de Grafică pe Calculator* – UPB, Automatică şi Calculatoare
2021-2022

Transformarea de vizualizare din spatiul 3D

- Are ca scop obtinerea unei vederi a unei scene 3D pe o suprafaţă de afişare.
- Poate fi comparată cu transformarea efectuată de o cameră foto asupra lumii reale pentru a se obţine imaginea dintr-o fotografie.
  - Imaginea obţinută depinde de:
    - poziţia camerei (observatorului)
    - orientarea camerei
    - ceea ce vede ochiul prin vizorul camerei foto (volumul vizual)
- Este o transformare compusă din mai multe transformări, între care şi o proiectie $R^3 \rightarrow R^2$
- Transformările se efectuează asupra vârfurilor obiectelor din scena 3D.
- Include şi operaţia de decupare (clipping) a primitivelor grafice la marginea volumului vizual (eliminarea părţilor din scena 3D care sunt în afara volumului vizual al vederii curente).

Transformarea varfurilor in OpenGL



Matricile transformarii varfurilor obiectelor 3D

M – matricea de modelare: difera de la obiect la obiect

V – matricea de vizualizare: aceeasi pentru intreaga scena vizualizata

P – matricea de proiectie: aceeasi pentru intreaga scena vizualizata

VM = V· M (matricea “model-view”)

MVP = P · V · M (matricea “model-view-projection”)

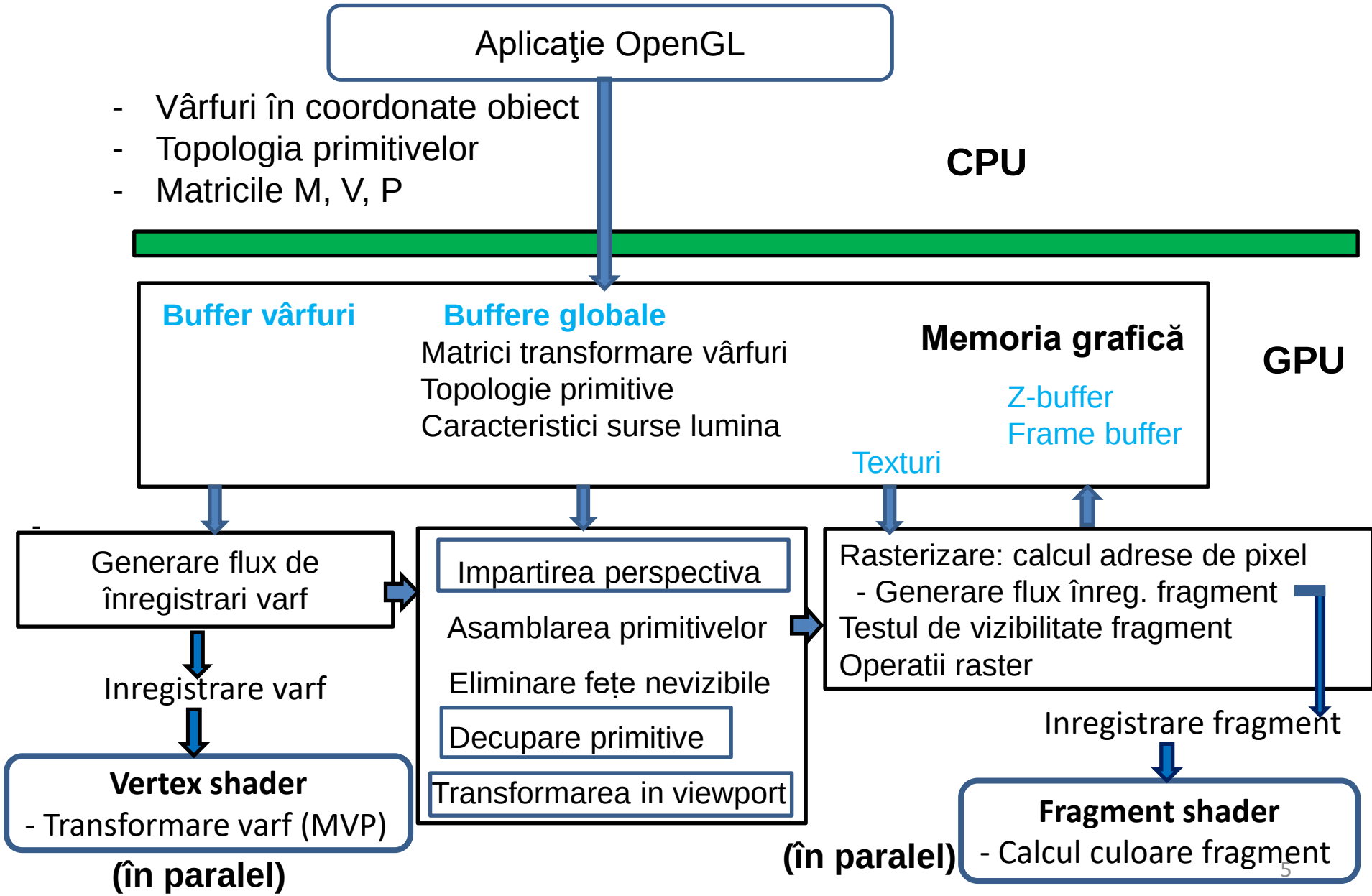
Matricile M, V, P sunt calculate în programul OpenGL folosind functii ale bibliotecii **glm** şi transferate la GPU.

Transformarea vârfurilor din coordonate obiect în coordonate de decupare (clip):

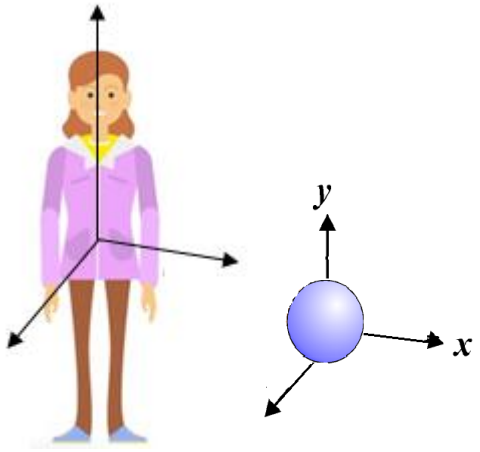
$$[xc, yc, zc]^T = PVM \cdot [xo, yo, zo]^T$$

- Transformarea vârfurilor este efectuată în programul ***Vertex Shader executat pe GPU,*** folosindu-se matricea MVP transferata din programul OpenGL.
- Programul ***Vertex Shader este scris de programator în GLSL*** (OpenGL Shading Language) şi transferat la GPU.

Banda grafica incepand cu OpenGl 3.0



Spatiul obiect



Spatiul obiect (Sistemul coordonatelor obiect)

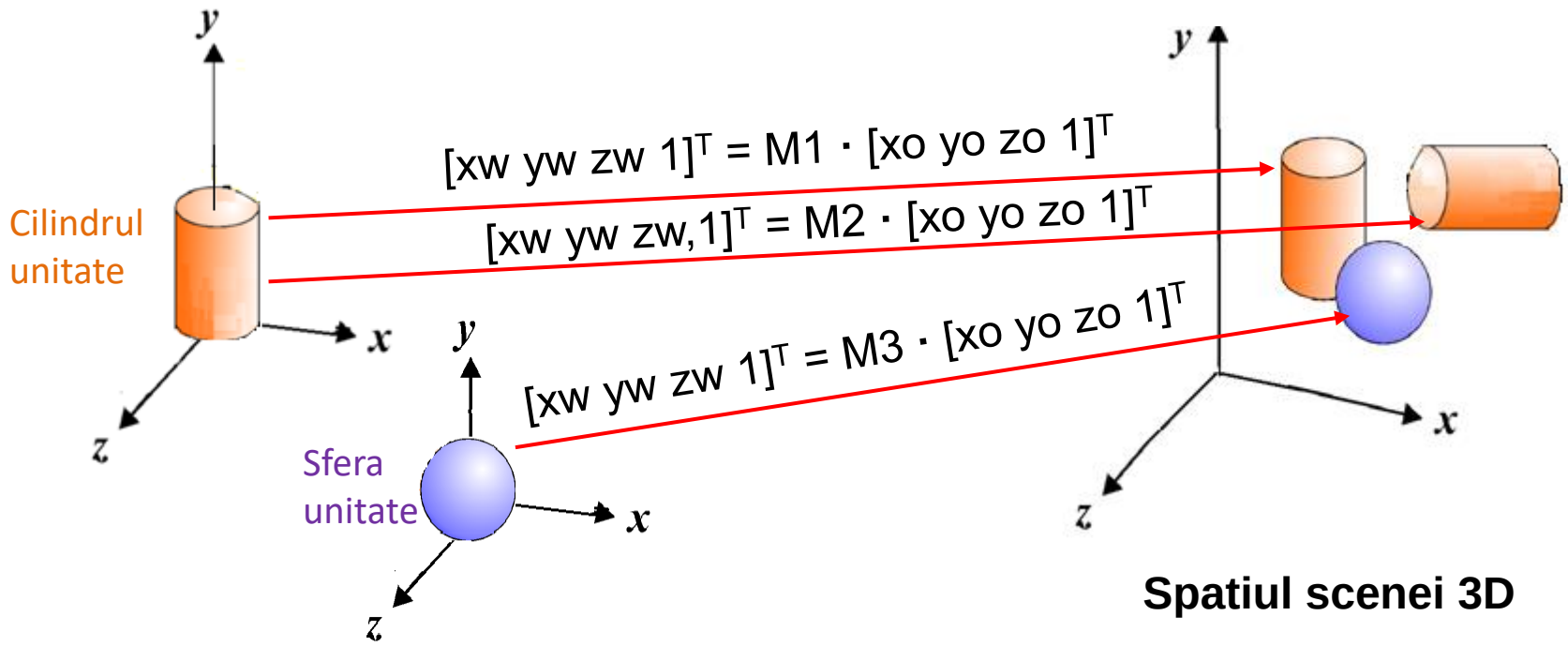
- In mod uzual, fiecare obiect este definit în propriul său sistem de coordonate (sistemul coordonatelor locale – coordonate obiect)
- Sistem de coordonate carteziene 3D « dreapta »
- Exemple: cubul, sfera, scheletul unui personaj animat, elemente de mobilier, etc.

  De regula, obiectul este definit centrat in originea sistemului sau de coordonate.
- Avantaje: usurinta modelarii şi reutilizarea modelelor obiectelor 3D.

Spatiul scenei 3D -World space (Sistemul coordonatelor globale)

- Spaţiul în care este compusă scena 3D
- Scena - raportată la un sistem de referinţă global – sistem de coordonate carteziene 3D “dreapta”, numit *Sistemul coordonatelor globale*
- Originea sistemului – aleasa de modelatorul scenei

Spatiul scenei 3D
Transformarea de modelare



Spatiul scenei 3D

Obiecte definite în sistemul coordonatelor globale

[xo yo zo 1]: coord omogene varf în sist de coord al unui obiect

[xw yw zw 1]: coord omogene varf în sist de coord globale

Transformarea de modelare:

Transformare geometrica compusa. Exemple:

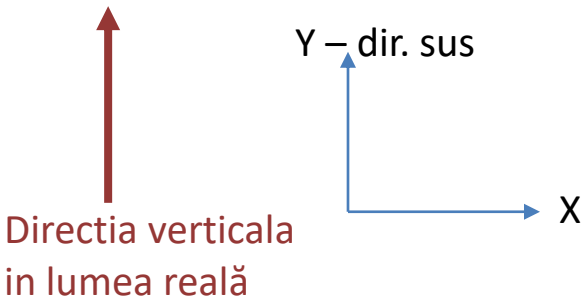
M1 = T(tx1,ty1,tz1) * S(sx,sy,sz)
M2 = T(tx2,ty2,tz2)* Roz (1.57) *S(sx,sy,sz)

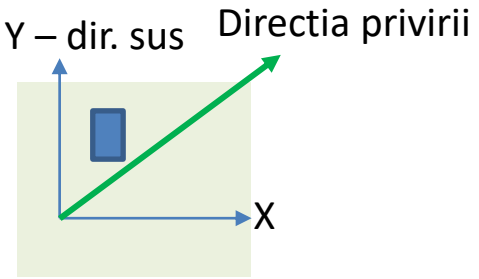
Spatiul camerei (observator)

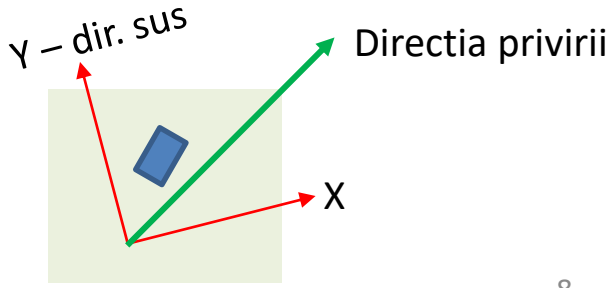
Analogie cu aparatul de fotografiat (camera reala)

Un aparat foto poate produce o fotografie in care apar o parte din obiectele spatiului real (3D).

- Continutul fotografiei depinde de:
  - pozitia aparatului in spatiul real
  - orientarea acestuia: directia in care priveste fotograful (observatorul)
  - rotatia aparatului față de pozitia sa normală
- Fotografia este formata intr-un plan 2D, care are atasat un sistem cartezian 2D, XOY:
- In pozitia normala a aparatului, directia "sus" a aparatului este directia axei OY (axa verticala).
- Directia in care priveste fotograful este perpendiculara pe planul imaginii, formand impreuna cu axele OX, OY un sistem de coord carteziene 3D.
- Fotograful poate roti aparatul de fotografiat, astfel incat directia "sus" a aparatului se schimbă.







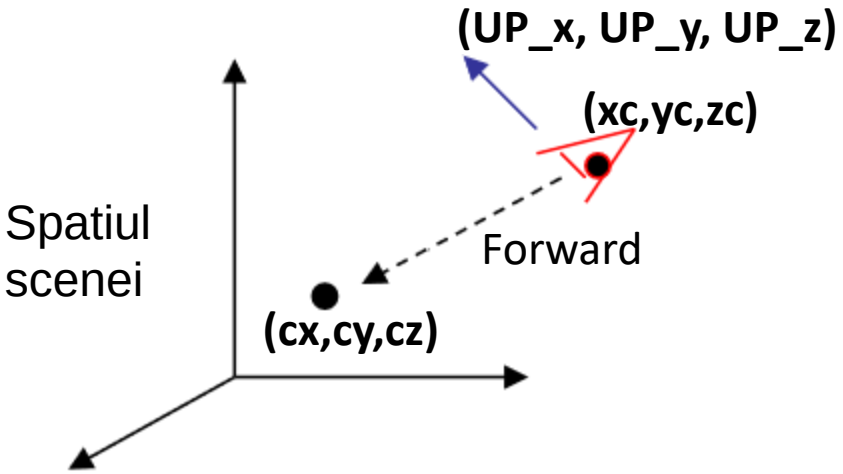
Spatiul camerei (observator)
Sistemul de coordonate al camerei

- Spaţiul camerei este spaţiul scenei 3D văzut din pozitia camerei virtuale (observatorului virtual) pe direcţia în care este orientata camera (priveşte observatorul virtual).
- Spatiul camerei este raportat la un sistem de coordonate atasat camerei – sistemul de coordonate al camerei (sistemul de coordonate observator).
- **Sistemul de coordonate al camerei este definit prin 3 parametri**:

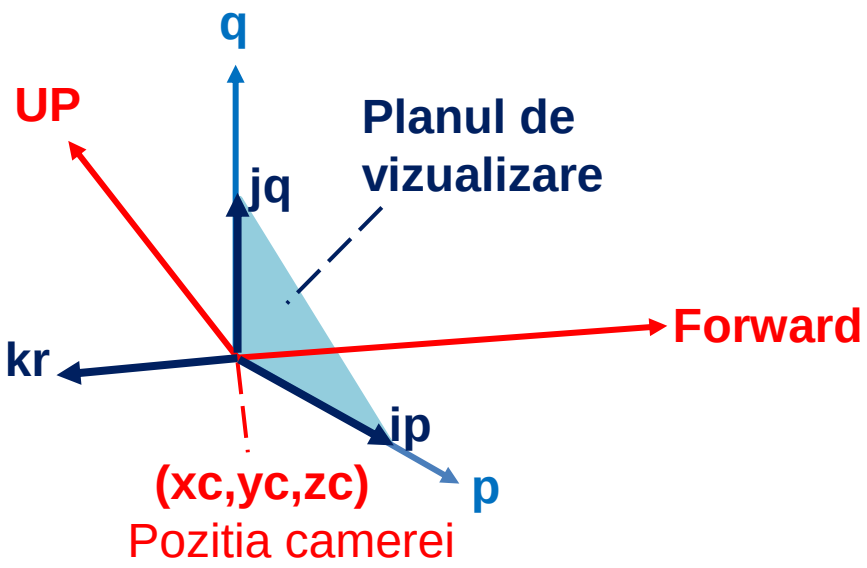
1. Originea sa: **pozitia camerei virtuale (observatorului virtual)** in spatiul scenei **(xc, yc, zc).**

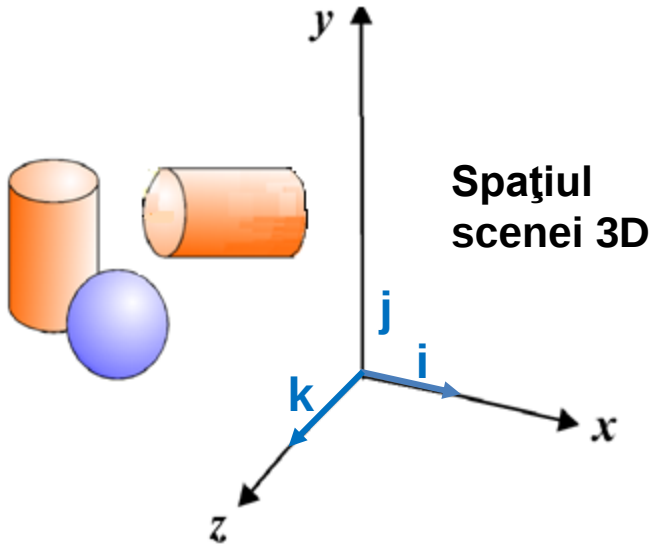
2. Directia in care priveste observatorul virtual (spre un punct din scenă) – **vectorul Forward**

3. **Directia "sus" a camerei** (**vectorul Up**): reprezinta **rotatia camerei**

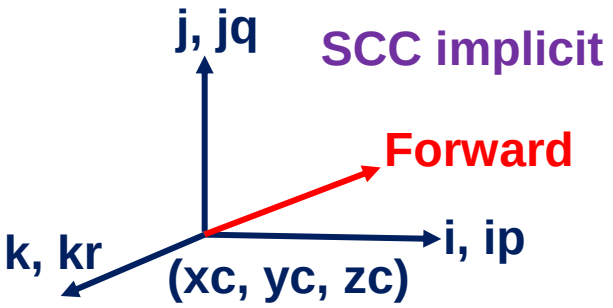


Sistemul de coordonate al camerei (SCC)



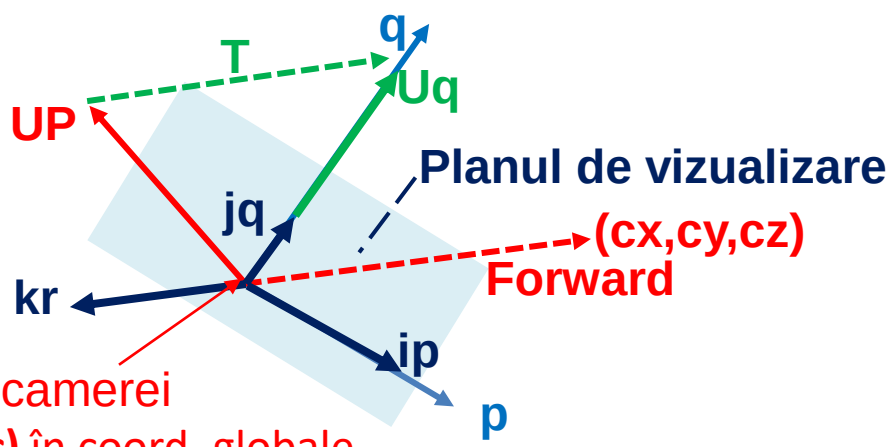


- (xc, yc, zc): originea sistemului de coordonate (pozitia camerei)
- (xc, yc, zc) impreuna cu vectorul **Forward** definesc un plan, numit **planul de vizualizare.**
- Forward – normala la planul de vizualizare
- axa q: axa verticală a planului de vizualizare
- axa p: axa orizontală a planului de vizualizare
- (ip,jq,kr) formeaza un sistem de coordonate 3D dreapta



ip, jq, kr: versorii axelor sistemului de coordonate al camerei coincid cu cei ai sist. coordonatelor globale

Determinarea axelor sistemului de coordonate al camerei (1)



Pozitia camerei
(xc,yc,zc) în coord. globale
(0,0,0) în coordonate cameră

kr = - Forward/|Forward|
Versorul directiei axei z a SCC

- Uq: proiectia ortogonala a vectorului UP in planul de vizualizare: Uq = UP + T
- Uq dă direcţia axei verticale a planului de vizualizare

T || (-kr) → T = s*(-kr) = - s*kr (s un scalar oarecare) → Uq = UP - s*kr

Uq perpendicular pe kr → Uq·kr = 0 → (UP - s*kr)·kr =0 → UP·kr - s*kr·kr = 0 → s=UP·kr

Uq = UP – (UP·kr)*kr

jq = Uq/|Uq| este versorul axei verticale a planului de vizualizare

ip = jq x kr /|jq x kr| este versorul axei orizontale a planului de vizualizare

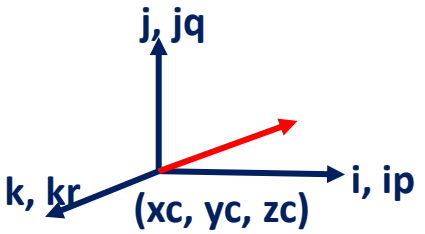
(ip, jq, kr) formeaza un sistem de coordonate carteziene 3D dreapta

Matricea transformarii de vizualizare (V) -1

Transformarea de vizualizare: **din spatiul scenei → în spatiul camerei**

$[xe\ ye\ ze\ 1]^T = V \bullet [xw\ yw\ zw\ 1]^T$ eye coordinates ← world coordinates

Sistemul de coordonate cameră implicit în OpenGL



- poziţia camerei: **originea sistemului de coordonate globale**
- Vectorul Forward: **direcţia negativa a axei OZ**
- Vectorul UP: **direcţia pozitivã a axei OY**

Cu aceste valori implicite, transformarea de vizualizare este transformarea identica.

Pentru calculul matricei transformarii de vizualizare se foloseşte functia **lookAt:**

glm::mat4 **V** =

glm::lookAt (glm::vec3 (xc,yc,zc), glm::vec3 (cx,cy,cz), glm::vec3 (UPx, UPy, UPz))

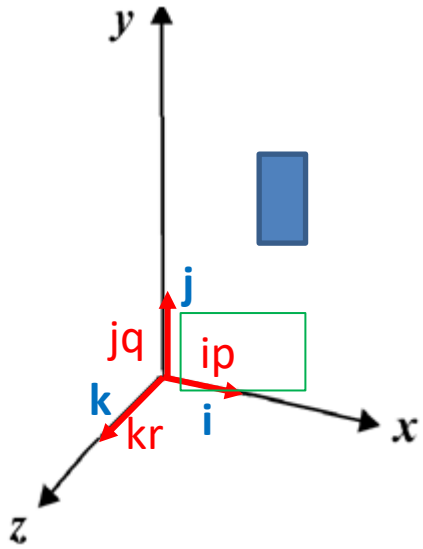
(xc, yc, zc): **poziţia camerei**

(cx,cy,cz): centrul de interes - punctul din scena 3D catre care priveste observatorul;

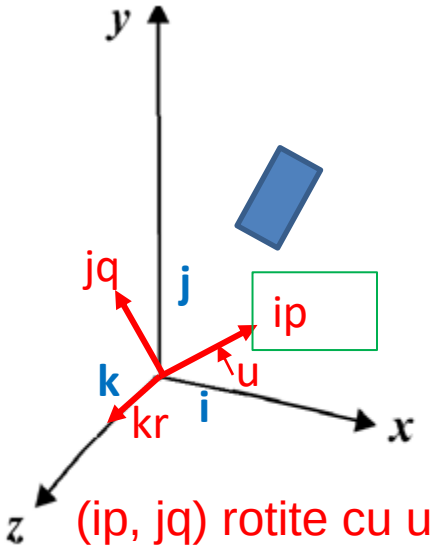
vectorul (xc,yc,zc) → (cx,cy,cz) reprezinta **vectorul Forward**

(UPx, UPy, UPz): v**ectorul « sus » al camerei (UP)**

Matricea transformarii de vizualizare -2



V = I (matricea identica)



(ip, jq) rotite cu unghiul u in jurul axei oz

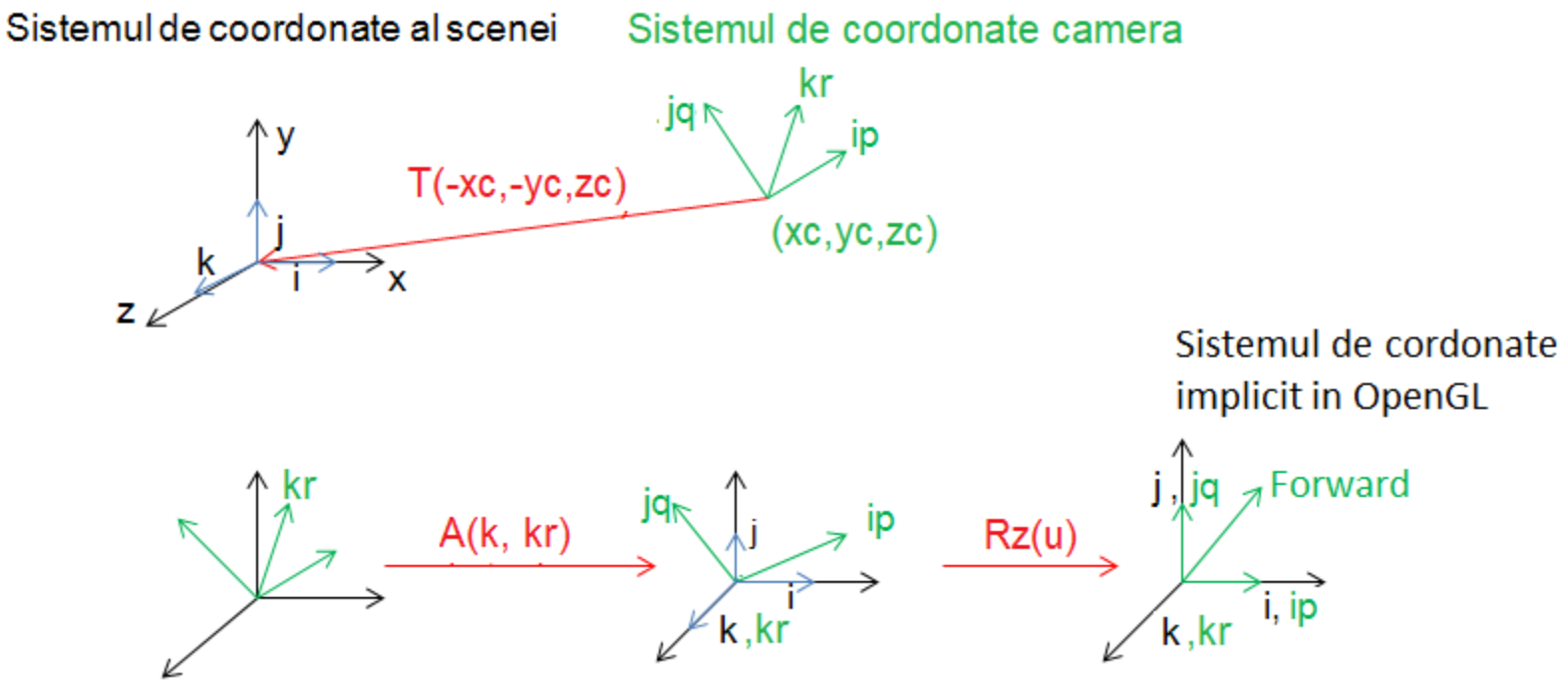
V?

V: matricea care transforma varfurile obiectelor din spatiul scenei in spatiul camerei (observator - eye) → $[xe\ ye\ ze\ 1]^T = R_{OZ}(-u) \bullet [xw\ yw\ zw\ 1]^T$

Matricea transformarii de vizualizare: $V = R_{OZ}(-u)$

- Aplicand V axelor sistemului de coordonate al camerei, se vor suprapune pe cele ale sistemului de coordonate globale.

Matricea transformarii de vizualizare -3



- A(k,kr) este o transformare prin care se aliniaza axele z ale celor 2 sisteme de coordonate: consta din 2 rotatii – o rotatie in jurul axei X si una in jurul axei Y.
- Prin rotatia in jurul axei z (Rz) se aliniaza celelalte 2 axe.

Matricea transformarii de vizualizare: V = Rz(u) * A(k,kr) * T(-xc,-yc,-zc)

Transformarea de modelare si vizualizare

din sistemul de coordonate obiect → in sistemul de coordonate observator

- Transformare compusa (model-view) : VM = V•M

  M – matricea de modelare folosita pentru transformarea unui obiect in spatiul scenei

  V – matricea de vizualizare a scenei

  Transformarea varfurilor din coordonate obiect in coordonate observator (*eye*):

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

- Pentru calculele de iluminare a obiectelor scenei 3D se folosesc vectori normali atasati varfurilor obiectelor.

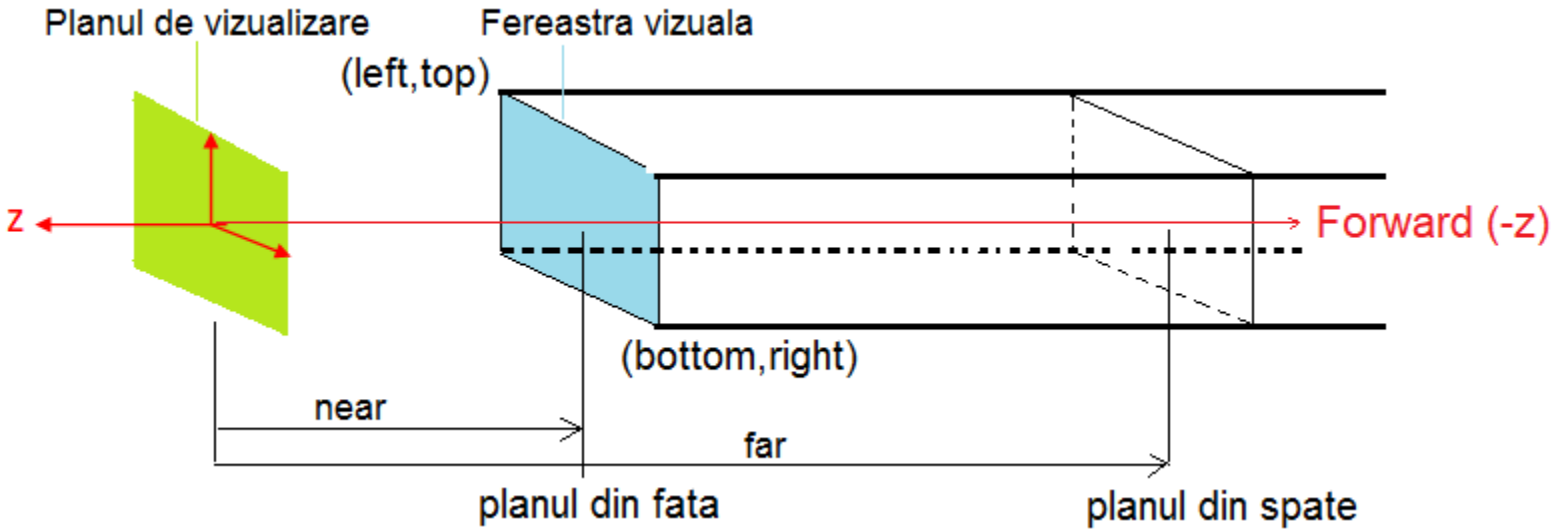
  Transformarea normalelor din coordonate obiect in coordonate observator:

  $[nxe\ nye\ \ nze\ we]^T = (VM^{-1})^T \bullet [nxo\ nyo\ nzo\ nwo]^T$

Transformarea de proiectie (1)

Determinata de **volumul vizual** definit de programator

Volumul vizual al proiectiei ortografice: paralelipiped dreptunghic cu laturile paralele cu axa z a sistemului de coordonate camera (SCC), delimitat in adancime de planul din fata (near) si planul din spate (far) al vederii; near si far sunt distanţe (pozitive) faţă de planul de vizualizare masurate pe axa z negativa a SCC.

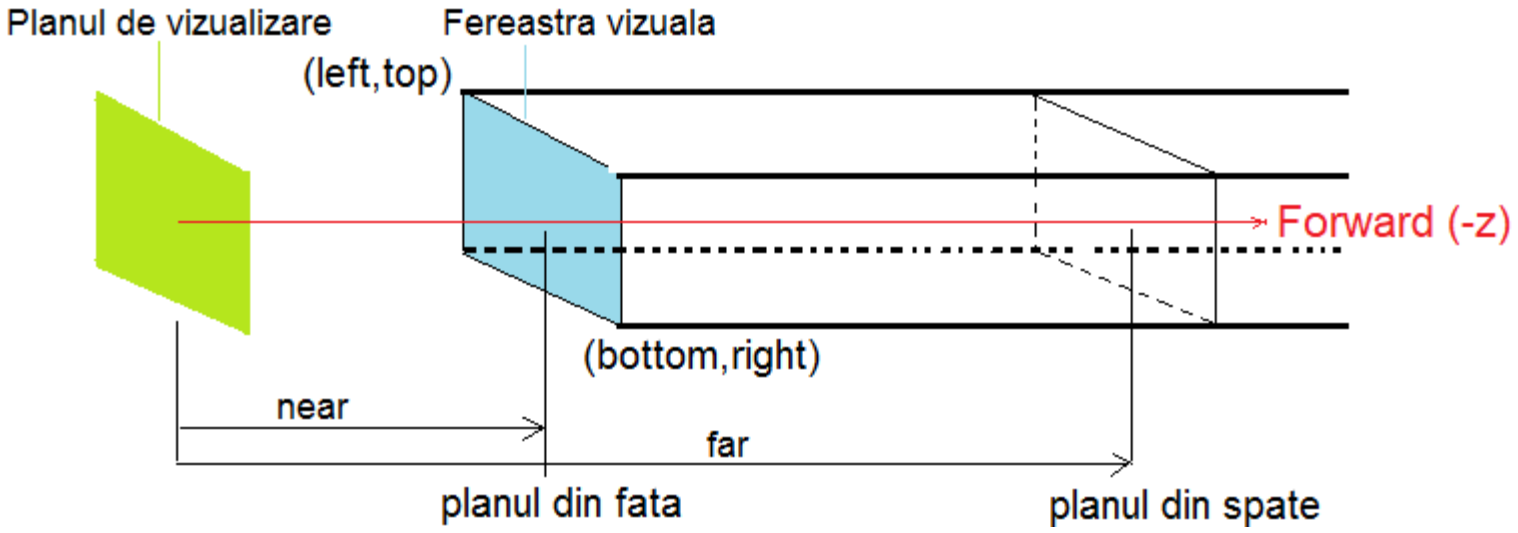


- Coordonatele colturilor ferestrei vizuale sunt raportate la sistemul de coordonate al planului din faţă, care are axele paralele cu cele ale planului de vizualizare.

Transformarea de proiectie (2)

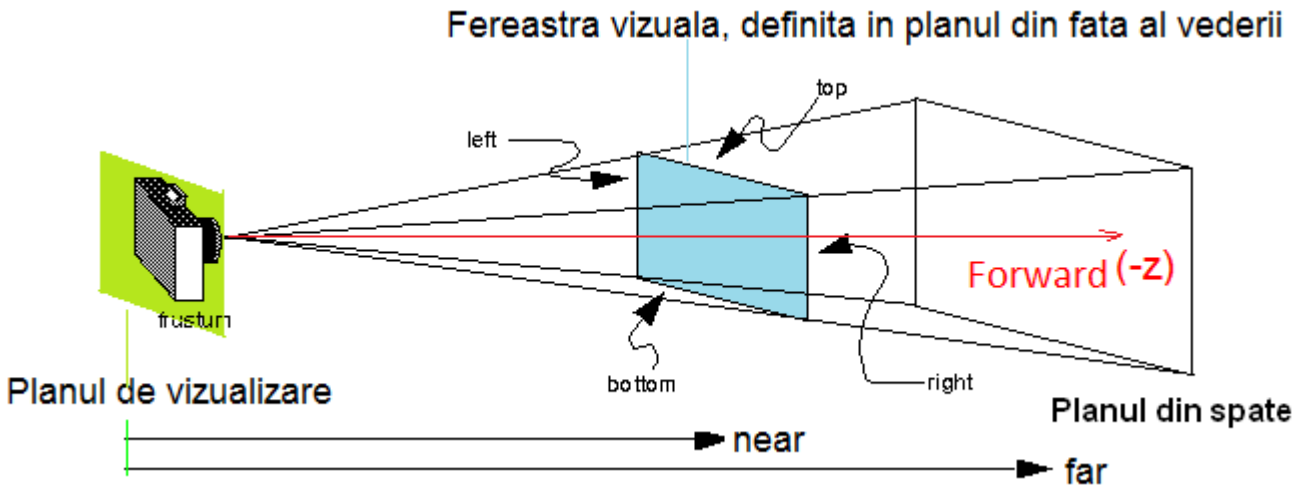
Proiectia ortografica

- Se efectueaza proiectia ortografica în fereastra vizuala a obiectelor cuprinse in volumul vizual. Primitivele grafice situate complet in afara volumului vizual sunt eliminate iar cele care intersecteaza volumul sunt decupate la frontiera volumului creându-se noi primitive.



Transformarea de proiectie (3)

Volumul vizual al proiectiei perspectiva (View frustum): trunchi de piramida delimitat in adancime de planul din faţă (near) si planul din spate (far) al vederii, cu varful in pozitia camerei.



Se efectueaza proiectia perspectiva în planul din faţă (in fereastra vizuala), cu centrul de proiecţie în poziţia camerei, a obiectelor cuprinse in volumul vizual. Obiectele situate complet in afara volumului vizual sunt eliminate iar cele care intersecteaza volumul sunt decupate la frontiera volumului creându-se noi primitive.

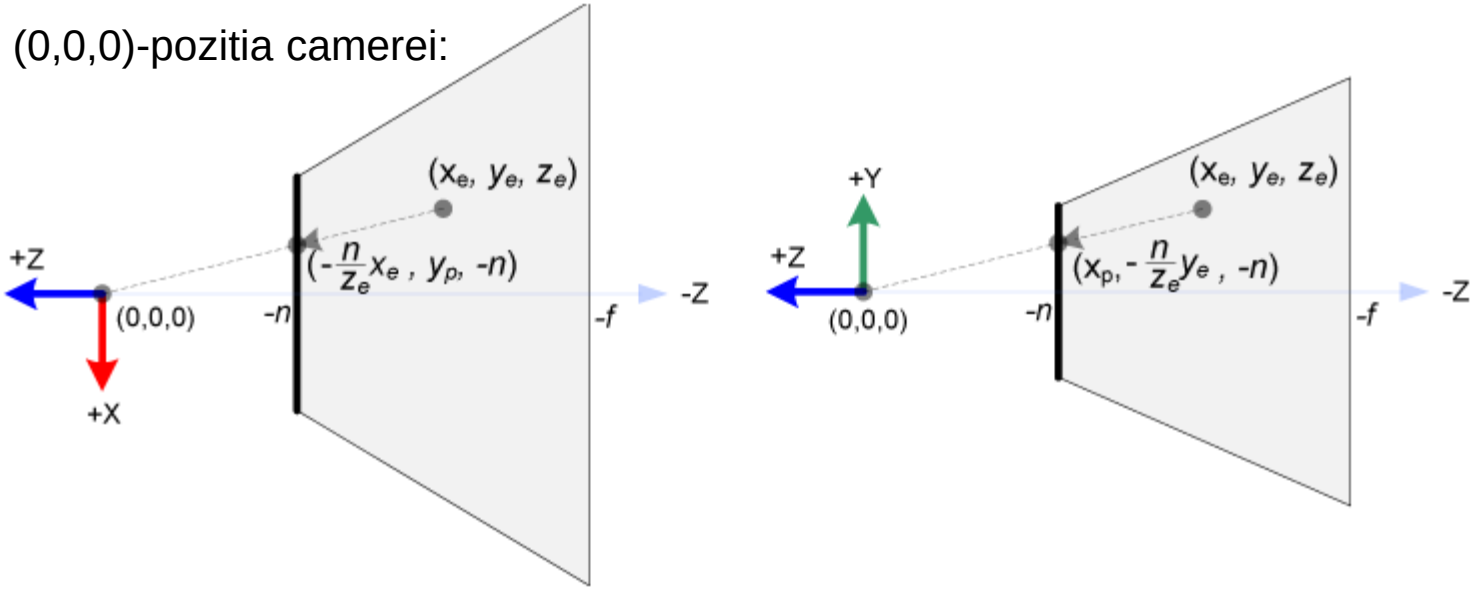
Transformarea de proiectie (4) - proiectia

- **Proiectia se efectueaza in fereastra vizuala din planul din faţă**: planul z =-near (in SCC).
- Varfurile proiectate sunt raportate la sist. coordonatelor cameră.

Proiectia ortografica a unui punct din sist. coordonatelor camera, (xe,ye,ze), in fereastra vizuala:

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

Proiectia perspectiva a unui punct din SCC, (xe,ye,ze), in planul z = -near, cu centrul de proiectie in (0,0,0)-pozitia camerei:



xp = (-near/ze)*xe, yp =(-near/ze)*ye – se pot obtine aplicand formulele proiectiei perspectiva cu centrul de proiectie in (0,0,0) si planul de proiectie z=-near.

Transformarea de proiectie (5) – Sistemul coordonatelor de decupare

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

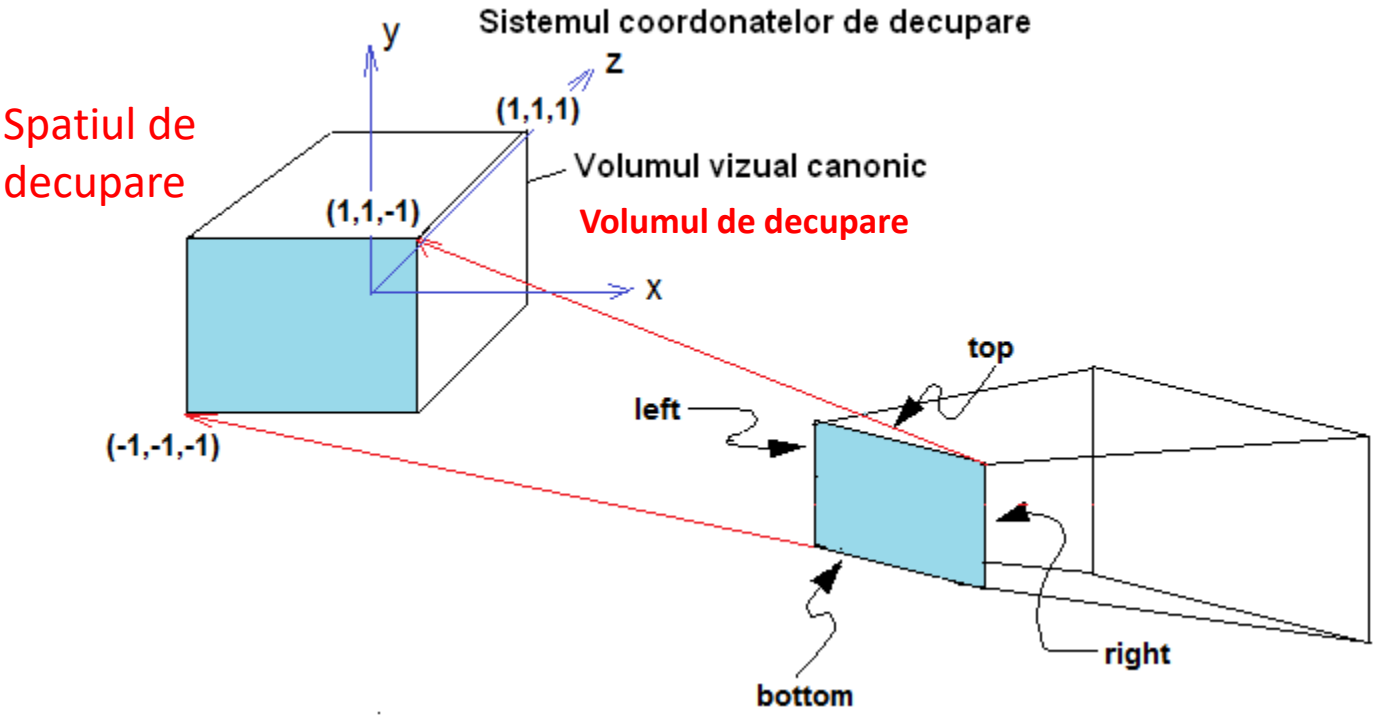
- In OpenGL transformarea de proiectie include: proiectia definita prin volumul de vizualizare compusa cu transformarea de normalizare.

  = transformarea coordonatelor camera in coordonate de decupare.

- Dupa transformarea varfurilor prin transformarea de proiectie, varfurile sunt unite pe baza topologiei definite de programator, formand primitive grafice 3D. Acestea sunt decupate la marginile volumului vizual canonic.

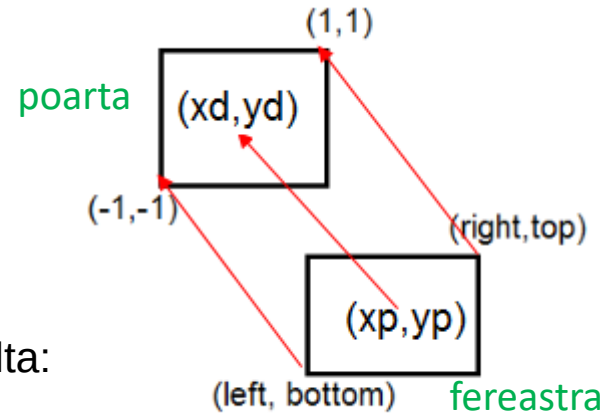
Transformarea de proiectie (6) – normalizarea

- *Sistemul coordonatelor de decupare*: sistem de coordonate carteziene 3D stanga.
- **Prin transformarea de normalizare:**
  - 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);
  - planul z= -near se mapeaza pe z= -1, iar z= -far se mapeaza pe z= 1



Transformarea de proiectie (7) calculul coordonatelor de decupare

- Impunand conditia (left,bottom) → (-1,-1) si (right,top) → (1,1), rezulta transformarea punctelor proiectate pe planul din faţă, (xp,yp), in sistemul coordonatelor de decupare – este o transformare freastra-poarta:



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

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

Pentru proiectia perspectiva:

Inlocuind xp si yp prin expresiile lor in functie de (xe,ye, near) 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)

(xd,yd,zd) sunt coordonate carteziene in sistemul coordonatelor de decupare (clip coordinates) ale varfurilor proiectate în fereastra vizuala din planul din față ;

Matricile transformarii de proiectie -1 eye coord.→ clip coord.

Pentru proiectia perspectiva:

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

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

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

$[xc\ yc\ \ zc\ wc]^T = Per \bullet [xe\ ye\ ze\ 1]^T$ xc = xd*wc, yc = yd*wc, zc=zd*wc, wc = -ze

(xd,yd,zd) – deduse anterior

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)

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

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

$[xc\ yc\ \ zc\ 1]^T = Par \bullet [xe\ ye\ ze\ 1]^T$

xc = xd, yc = yd, zc=zd

Matricile transformarii de proiectie -2

Matricile de transformarii de proiectie pot fi obtinute apeland urmatoarele functii din biblioteca glm:

glm::ortho(float left, float right, float bottom, float top, float zNear, float zFar);

glm::frustum (float left, float right, float bottom, float top, float zNear, float zFar);

Exemplu:

```
glm::mat4 PerspProjectionMatrix = glm::frustum(2, 100, -2, 100, 0.1f, 1000.f);
```

glm::perspective (float fov, float aspect, float znear, float zfar);

Creaza o matrice de proiectie pentru un volum perspectiva simetric:

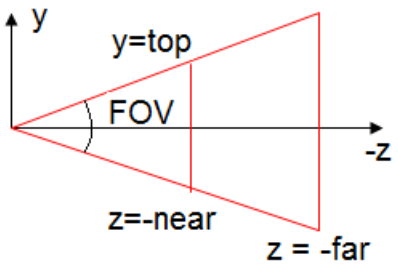
- fov: Field of View – unghiul de vizualizare pe orizontala in grade

tan(FOV/2) = top/near

- aspect: aspect ratio

aspect ratio = (right-left)/(top-bottom)

= raportul dintre latimea si inaltimea ferestrei vizuale



Impartirea perspectiva

Transformarea de proiectie: coordonate camera → coordonate de decupare

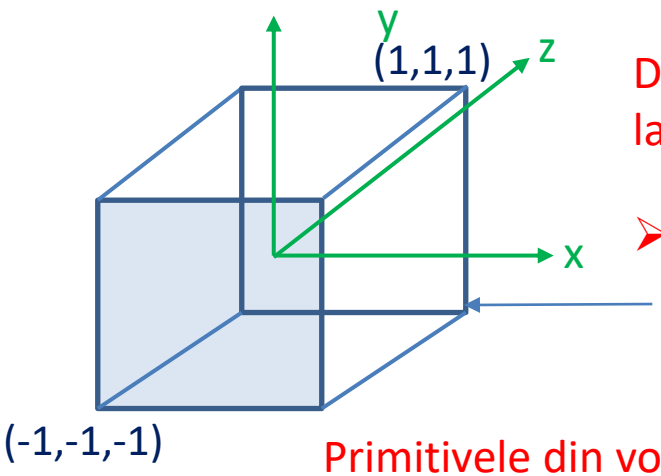
$[xc\ yc\ zc\ wc]^T$ = Par/Per • $[xe\ ye\ ze\ 1]^T$ (clip coordinates) ← eye coordinates)

Impartirea perspectiva: coordonate de decupare → coordonate dispozitiv normalizate

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

- **Varfurile rezultate din împartirea perspectiva sunt asamblate în primitive grafice care sunt decupate la frontiera volumului vizual canonic.**

Sistemul coordonatelor de decupare



După transformarea de proiectie, observatorul (camera) este la infinit pe axa OZ negativă a sistemului coord. de decupare.

- Distanța față de observator se masoara pe axa OZ pozitivă.

Volumul de decupare

După decupare, coord. primitivelor: -1 <= xd, yd,zd <=1

Primitivele din volumul de decupare sunt proiectate ortografic pe fața "din față" a volumului: (-1,-1,-1) – (1,1,-1).

Transformarea în poarta de afisare
coordonate dispozitiv normalizate → coordonate ecran

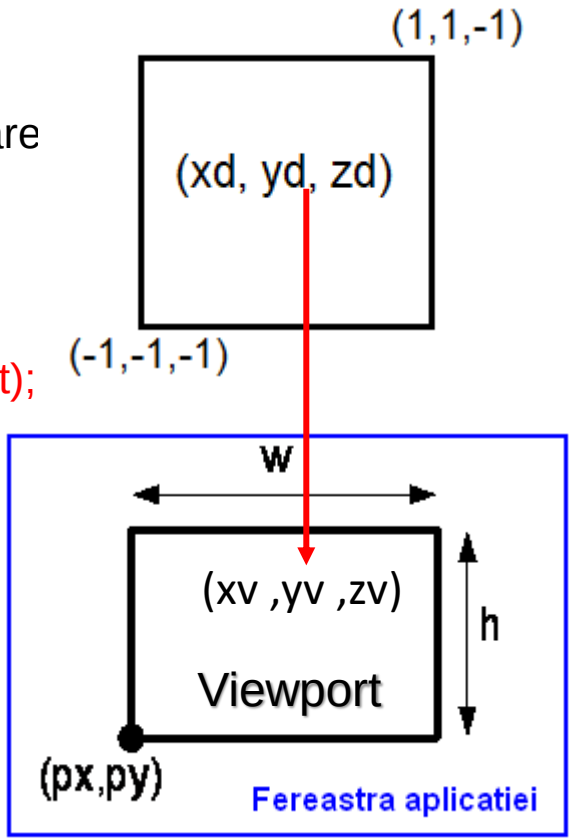
- Este o transformare fereasta-poarta, in care:
- Fereastra este fereastra 2D din sistemul coordonatelor de decupare

  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 coordonate relative la coltul stanga-jos al ferestrei aplicatiei (în pixeli). Valorile implicite : (0,0).

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



- **Transformarea în poarta de afisare este definita astfel:**

  xv = (width/2)xd + ox , ox = px+ width/2

  yv = (height/2)yd + oy, oy = py + height/2

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

  n (near) şi f (far) au valorile implicite 0.0 respectiv 1.0, pentru care 0<=zv<=1

  n si f pot fi modificate la valori cuprinse în intervalul [0,1] folosind funcţia ***DepthRange()***

Funcții glm utile în specificarea transformarii varfurilor

Trasformarea de modelare:

glm::mat4 M = produs de matrici de transformare geometrica

Transformarea de vizualizare:

glm::mat4 **V** = glm::lookAt (glm::vec3 (xc,yc,zc), glm::vec3 (cx,cy,cz), glm::vec3 (UPx, UPy, UPz))

Transformarea de proiectie:

glm::mat4 **P** = glm::ortho(float left, float right, float bottom, float top, float zNear, float zFar);

glm::mat4 **P** = glm::frustum (float left, float right, float bottom, float top, float zNear, float zFar);

glm::mat4 **P**= glm::perspective (float fov, float aspect, float znear, float zfar);

Transformarea în poarta de afişare:

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