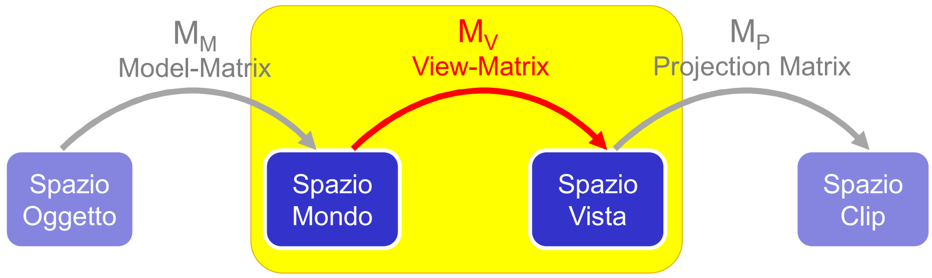
Trasformazioni nel pipeline di rendering visione di insieme



40

Matrice di vista

- dipende dalla dove *"camera"* (macchina fotografica)
  - detta anche:
    - pos del viewer
    - eye position
    - PoV (Point of View)

41

Spazio (sistema di riferimento) Vista

- ✓ O eye-space, o view-space, o camera-space
- ✓ Un sistema basato (e orientato) come l'osservatore
- ✓ Definizione degli assi e origine:
  - ⇒ origine: nel centro focale della macchina fotografica
  - ⇒ asse X: verso la destra della macchina fotografica
  - ⇒ asse Y: verso l'alto della macchina fotografica
  - ⇒ asse Z: verso la scena
    o verso il fotografo, nei sistemi a mano destra
    (comunemente adottato in OpenGL)
- ✓ Es: come cambia lo spazio vista se passo da una inquadratura verticale a una portrait (e «in bolla»)?

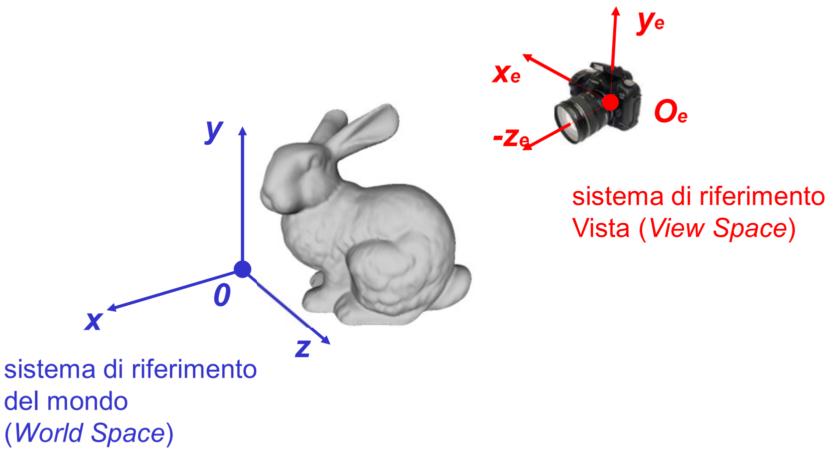
42

Trasformazione (matrice) di vista

- ✓ Da spazio Oggetto a Spazio Vista
- ✓ Definisce come viene *inquadrata* la scena
  - ⇒ cioè dove è piazzata e come è orientata la macchina fotografica virtuale (la «camera») che la riprende
  - ⇒ dipende cioè dai cosidetti
    parametri ESTRINSECI della camera
- ✓ Tipicamente è una trasformazione rigida
- ✓ Ricordare: le trasformazioni si possono combinare!
  - ⇒ Model-View matrix: matrice di modellazione-vista
    da spazio oggetto a spazio a spazio Mondo
  - ⇒ Come si ottinene?

43

Spazio (sistema di riferimento) Vista



44

Trasformazione di "Vista"

«un computer graphicist preferisce portare la montagna davanti alla macchina fotografica, piuttosto che la macchina fotografica davanti alla montagna»

- ✓ Da: **World Frame**
  A: **View Frame**
- ✓ Dipende interamente dai «**parametri estrinseci**» della macchina fotografica (virtuale)
  - ⇒ Cioè da *dove è, e come è orientata* (nel mondo)
  - ⇒ (per es: un tipico task di Computer Vision: «registrare una foto» = evincere i parametri estrinseci della camera al momento del suo scatto)
- ✓ E' un cambio di sistema di riferimento, cioè una trasformazione affine…
  - ⇒ Matrice di Vista = la Matrice che fa questa trasformazione

45

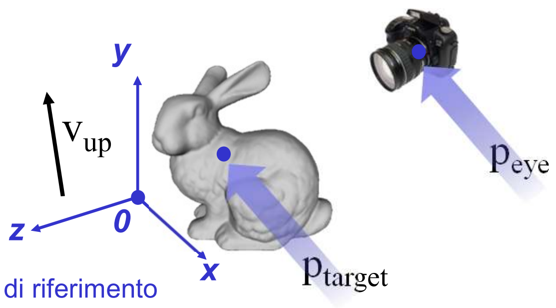
Esempio tipico di costruzione transformazione di vista

Input:

1) camera position: p_{eye}
2) target position: p_{target}
3) vettore di alto: v_{up}

nb: punti e vettori espressi in spazio mondo!

un esempio di descrizione esaustiva dei *parametri estrinseci* della camera



sistema di riferimento globale *(world frame)*

46

Esempio tipico di costruzione transformazione di vista

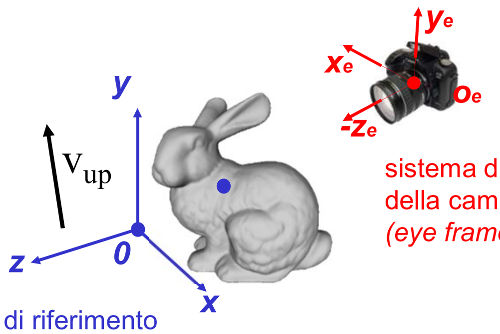
Input:

1) camera position: p_{eye}
2) target position: p_{target}
3) vettore di alto: v_{up}

Output:

Matrice di Trasformazione

world frame ⟶ *eye frame*



sistema di riferimento della camera *(eye frame)*

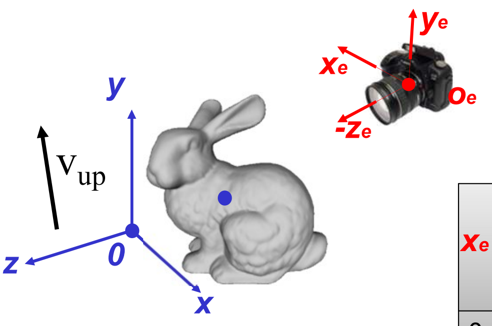
sistema di riferimento globale *(world frame)*

47

Esempio tipico di costruzione transformazione di vista

Input:

1) camera position: p_{eye}
2) target position: p_{target}
3) vettore di alto: v_{up}




```
vec3 oe;
vec3 xe, ye, ze;

ze = p_target - p_eye;
ze = -ze;
ze = normlize( ze );

xe = cross( vup , ze );
xe = normalize( xe );

ye = cross(ze, xa);
```

$$\begin{bmatrix} x_e & y_e & z_e & o_e \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

matrice che va da spazio vista a spazio mondo.
E' l'inversa di quella che volevamo.
Ergo, va invertita.

48

Esempio tipico di costruzione transformazione di vista

Origine e assi del sistema vista espressi nelle coord del sistema mondo

L'asse zeta va verso l'osservatore

Deve essere reso unitario

"completamento di base"

normalizzaz non necessaria (perché?)

```
vec3 oe;
vec3 xe, ye, ze;

ze = p_eye - p_target;
ze = normlize( ze );

xe = cross( vup , ze );
xe = normalize( xe );

ye = cross(ze, xa);
```

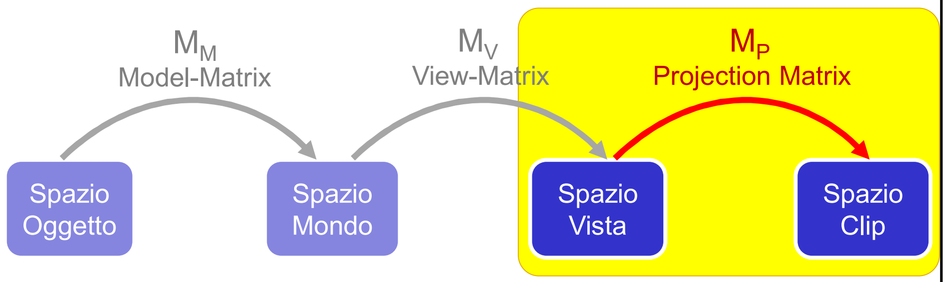
nb: quando si fallisce? le due normalizz possono essere div by 0? quando?

$$\begin{bmatrix} x_e & y_e & z_e & o_e \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

matrice che va da spazio vista a spazio mondo.
E' l'inversa di quella che volevamo.
Ergo, va invertita.

49

Trasformazioni nel pipeline di rendering visione di insieme

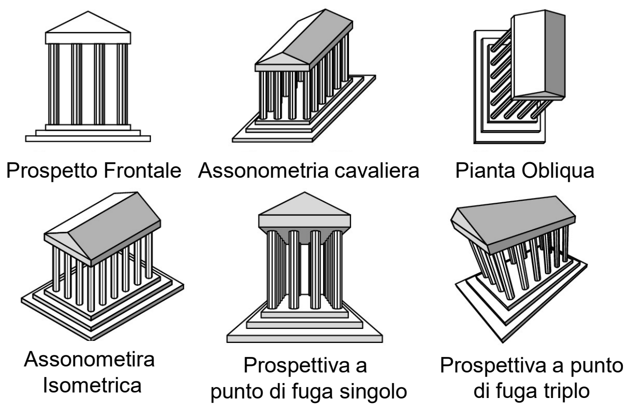


58

Trasformazione di proiezione

- ✓ Vecchio problema:
  - ⇒(in arte, architettura progettazione)
- ✓ come riportare
  - ⇒oggetti 3D
  - ⇒su un piano 2D

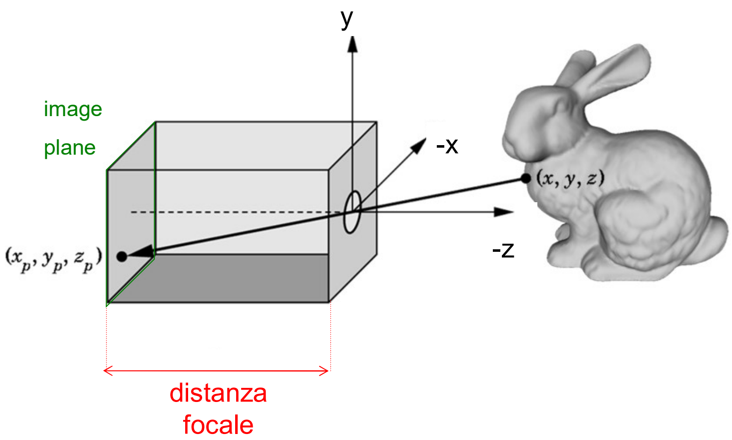
Alcune soluzioni classiche:



62

Nostro modello semplificato:

✓ pin-hole camera



68

Parametri *intrinseci* della camera: (quelli principali)

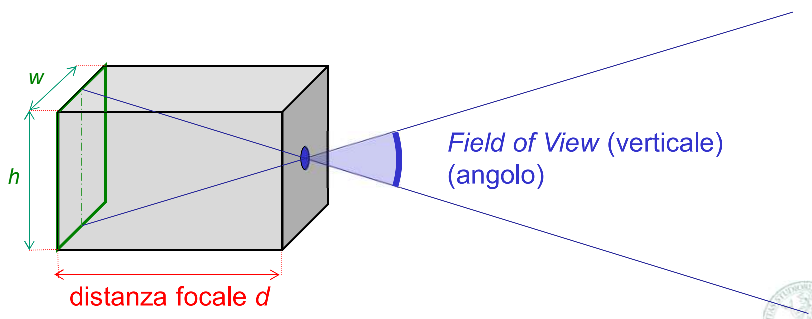
✓ dim image plane (w , h) (in spazio vista)

✓ distanza focale (d) (in spazio vista) oppure Field of View (FoV)

⇒ (come ottenere uno dall'altro?)

⇒ d si usa nei conti, ma FoV è intuitivo da settare, :

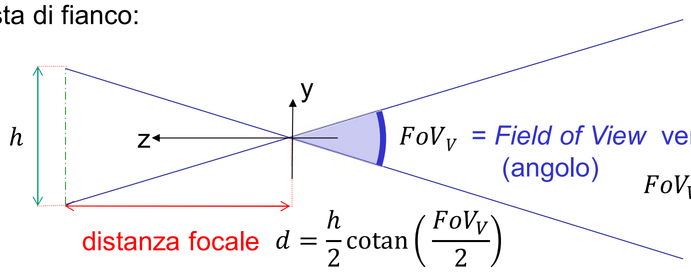
- FoV grande >60° (dist foc. piccola): «grandangolo»
- FoV piccolo <45° (dist foc. grande): «teleobiettivo»



69

Da Field of View a Distanza focale

Vista di fianco:

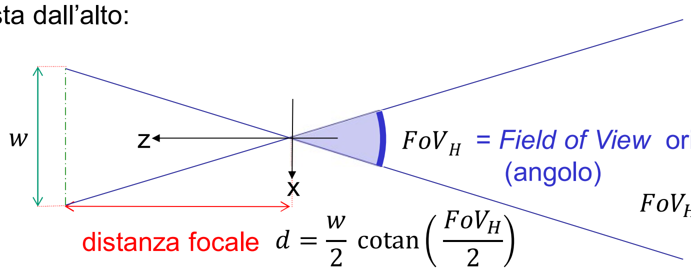


FoV_V = *Field of View* verticale (angolo)

$$FoV_V = 2 \arctan\left(\frac{h}{2d}\right)$$

distanza focale $d = \frac{h}{2} \text{cotan}\left(\frac{FoV_V}{2}\right)$

Vista dall'alto:



FoV_H = *Field of View* orizzontale (angolo)

$$FoV_H = 2 \arctan\left(\frac{w}{2d}\right)$$

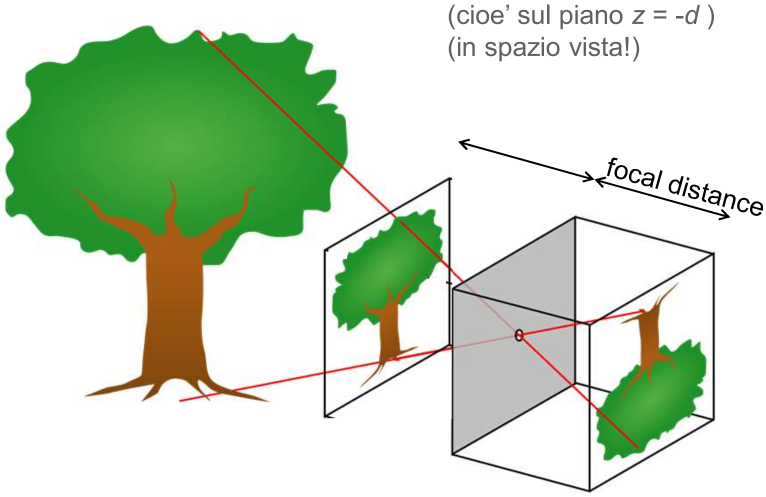
distanza focale $d = \frac{w}{2} \text{ cotan}\left(\frac{FoV_H}{2}\right)$

70

Pin Hole camera

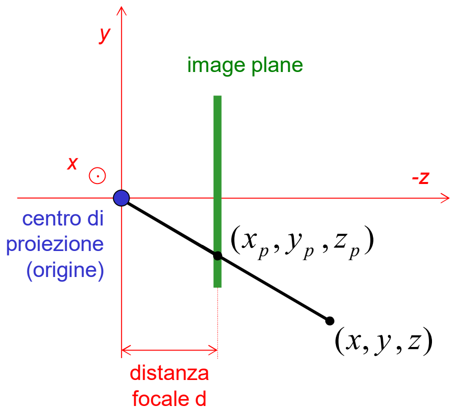
per semplicità, immaginiamoci il piano immagine *davanti* alla camera (e nn ribaltato), piuttosto che·· dietro (e ribaltato).

(cioe' sul piano $z = -d$)
(in spazio vista!)



71

Matematicamente



$$\begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = k \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$ con k t.c. $z_p = -d$

quindi…

$k = -d/z$ e $$\begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = \begin{pmatrix} -d \cdot x/z \\ -d \cdot y/z \\ -d \end{pmatrix}$$

Nota:
non è affine;
Per es, non mantiene parallelismo
(ma mantiene: colinearità)

72

Estendiamo la rappresentazione di punti e vettori in coordinate **omogenee**

Punti:

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

1 ➡

Vettori:

$$\vec{v} = \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix}$$

0 ➡

74

Estendiamo la rappresentazione di punti e vettori in coordinate **omogenee**

Punti:

$$\neq 0 \rightarrow \quad \mathbf{p} = \begin{bmatrix} x \\ y \\ z \\ * \end{bmatrix}$$

Vettori:

$$0 \rightarrow \quad \vec{v} = \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix}$$

75

Estendiamo la notazione

✓ Esprimo i *punti* anche con la notazione

$$\mathbf{p} = \begin{bmatrix} wx \\ wy \\ wz \\ w \end{bmatrix} \quad \text{con} \quad w \neq 0$$

$$\begin{bmatrix} wx \\ wy \\ wz \\ w \end{bmatrix} \xrightarrow{\text{divisione per 4ta comp}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

anche detta normalizzazione affine

76

Estendiamo la notazione

- Per es:

$$\begin{bmatrix}20\\10\\30\\10\end{bmatrix} \begin{bmatrix}2\\1\\3\\1\end{bmatrix} \begin{bmatrix}0.2\\0.1\\0.3\\0.1\end{bmatrix} \begin{bmatrix}4\\2\\6\\2\end{bmatrix} \cdots$$

sono alcune delle **coordinate omogenee** del **punto** che ha le seguenti **coordinate cartesiane**

$$\begin{bmatrix}2\\1\\3\end{bmatrix}$$

punto

questa è in "forma normale" ($w = 1$)

$$\begin{bmatrix}2\\1\\3\\0\end{bmatrix}$$

sono le uniche **coordinate omogenee** del **vettore** che ha le seguenti **coordinate cartesiane**

$$\begin{bmatrix}2\\1\\3\end{bmatrix}$$

vettore

77

Coordinate omogenee di punti

- Il punto P di coordinate cartesiane (x,y,z) è rappresentato in coordinate omogenee come (xw,yw,zw,w), con w qualunque (ma non 0)
- Set di coordinate omogenee diversi (x, y, z, w) e (x', y', z', w') possono rappresentare lo stesso punto;
  - ⇒ quando?
- Quando $w = 1$ (forma *canonica*) le coord cartesiane del punto coincidono con le prime tre coord omogenee.
- Con $(x, y, z, w \neq 0)$ si rappresentano *punti*, con $(x, y, z, 0)$ si rappresentano*vettori.*
- *Nota: tutte le matrici di trasformazione viste fin'ora funzionano anche con questa notazione generalizzata! Es: VERIFICA.*

78

Coordinate omogenee (o affini)

- ✓ Tutte le matrici di trasformazione che abbiamo visto fin'ora continuano a funzionare anche con i punti espressi in coordinate omogenee non canoniche (w!=1)
- ✓ Esempio, traslazione:

$$\begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} 20 \\ 15 \\ 50 \\ 10 \end{pmatrix} = \begin{pmatrix} 70 \\ 35 \\ 80 \\ 10 \end{pmatrix}$$

Matrice di traslazione del vett (+5 , +2 , +3)

Punto di coord cartesiane (2 , 1.5 , 5)

Punto di coord cartesiane (7 , 3.5 , 8)

(+5 , +2 , +3)

- ✓ Ma ora possiamo esprimere anche la proiezione prospettica…

79

Matrice di proiezione prospettica

matrice di trasformazione per la **proiezione prospettica** (nota: non è affine)

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix}$$

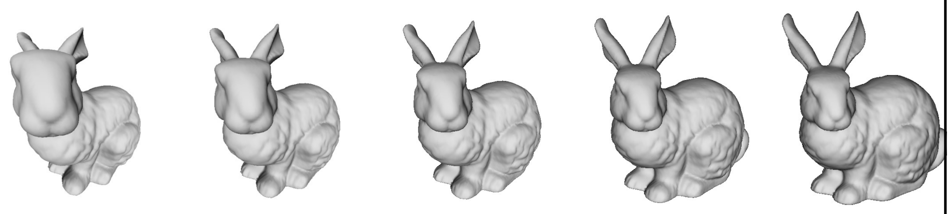
$$\mathbf{P}\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ -z/d \end{pmatrix}$$

…che esprime il punto di coordinate *cartesiane…* (divido per w)

$$\begin{pmatrix} -x \cdot d/z \\ -y \cdot d/z \\ -d \end{pmatrix}$$

80

Proiezione Prospettica: che effetto fa



d piccolo

d grande

d infinito
(diventa
una proiezione
ortogonale)

Più distorsione
prospettica.

Effetto "fish-eye"
(grandangolo)

Proporzioni
più mantenute

Effetto "zoom"
(eg. vista dal
satellite)

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix}$$

83