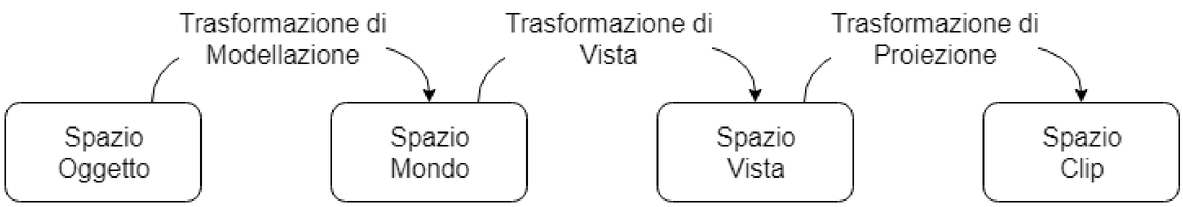
Pipeline di trasformazione

Overview

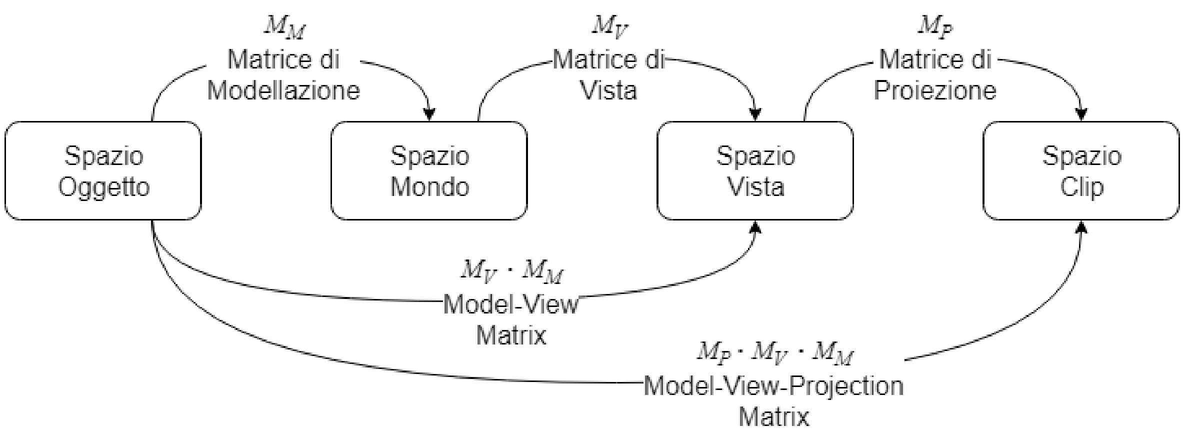
Usando le trasformazioni spaziali, possiamo ora cercare di studiare il problema all'interno del pipeline di rendering rasterization based di trasformare il modello 3D in un'immagine 2D da rasterizzare.

Iniziamo a formare una visione d'insieme del problema.
Dobbiamo trasformare le coordinate dei vertici di un modello 3D in coordinate 2D dello spazio clip, quindi passare dallo **spazio oggetto** allo **spazio clip**.

Per farlo applicheremo una matrice di trasformazione, ottenuta come composizione di 3 trasformazioni, ciascuna che ci fa passare da uno spazio a un altro e da un sistema di coordinate a un altro:



Intendendo le trasformazioni come matrici, avremo:

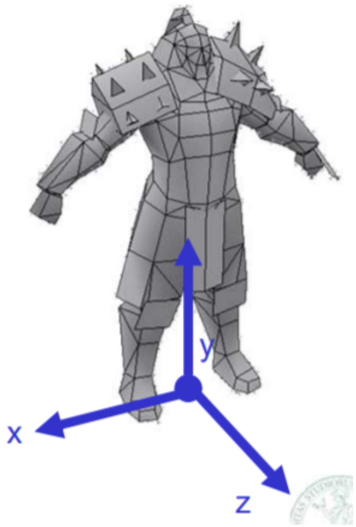


Gli spazi

Prima di studiare ciascuna matrice, vediamo prima gli spazi.

Spazio Oggetto

Lo **spazio oggetto** è il sistema di riferimento in cui è definito ciascun modello 3D.
Ad esempio le coordinate dei vertici di una mesh o le coordinate dei punti di controllo di una patch di Bézier sono espresse in spazio oggetto. La definizione dello spazio oggetto è una scelta arbitraria di chi costruisce il modello o del software sui cui è costruito.



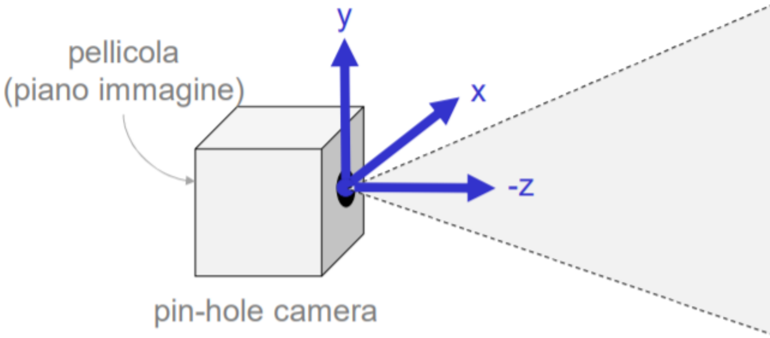
Spazio Mondo

Lo **spazio mondo** è il sistema di riferimento in cui è definita tutta la scena 3D.
Ad esempio, data una scena "sala da ballo", in questo spazio vi sono le coordinate degli oggetti rispetto al sistema di riferimento della sala da ballo, la quale magari ha come origine il centro della sala, come asse x la direzione Est-Ovest etc.
Anche questo spazio è definito in modo arbitrario da chi costruisce la scena o dal software.

Spazio Vista

Lo **spazio vista** è il sistema di riferimento in cui la scena è definita rispetto alla posizione della *camera*.
In questo spazio, tutti gli elementi della scena sono quindi definiti in relazione alle coordinate che hanno come origine la camera. In dettaglio, le coordinate di questo spazio sono le seguenti:

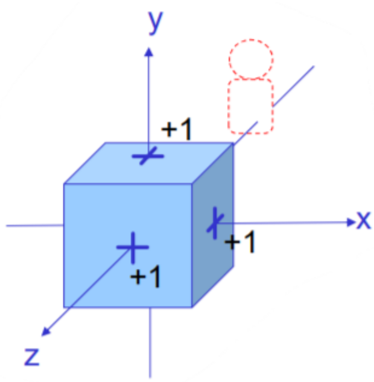
- *Origine* : POV della camera
- *asse x* : asse orizzontale della camera
- *asse y* : asse verticale della camera
- *asse z* : asse che va dal davanti al dietro della camera



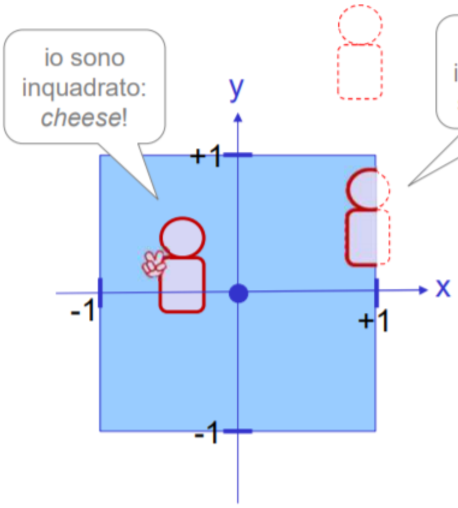
Spazio Clip

Lo **spazio clip** è il sistema di riferimento dove l'immagine inquadrata nello spazio vista viene inserita in uno spazio di coordinate standard che è allineato allo schermo e che sarà allineato anche all'immagine finale prodotta dal rendering. Questo spazio ha le seguenti coordinate:

- *Origine* : centro dello schermo/immagine
- *asse x* : asse orizzontale allo schermo/immagine, da sinistra (-1) a destra (+1)
- *asse y* : asse verticale allo schermo/immagine, dal basso (-1) verso l'alto (+1)
- *asse z* : asse ortogonale allo schermo/immagine, che va dal davanti (-1) al dietro dell'osservatore (+1)



Visto allineato allo schermo, come lo si dovrebbe immaginare, lo spazio clip sarà:



Come si nota, solo gli elementi della scena che hanno coordinate comprese tra -1 e +1 in tutti e tre gli assi sono inquadrati. Inoltre le coordinate dello spazio clip sono indipendenti dal monitor in cui verrà visualizzato il rendering, per questo sono anche dette **NDC** (***N**ormalized **D**evice **C**oordinates*).

Le trasformazioni - overview

Trasformazione di Modellazione

Trasforma i punti/vettori da **spazio oggetto** a **spazio mondo**, riflettendo come sono disposti gli oggetti nella scena.
Ogni oggetto ha la sua trasformazione di modellazione e quando un oggetto cambia posizione nella scena, cambia la sua trasformazione di modellazione.

Trasformazione di Vista

Trasforma i punti/vettori da **spazio mondo** a **spazio vista**, riflettendo la posizione della camera che inquadra la scena (*parametri estrinseci* della camera).
Se la camera si sposta, cambia la trasformazione di vista, causando uno spostamento nell'immagine degli oggetti inquadrati, anche se la loro posizione, e quindi la loro matrice di modellazione non cambia.

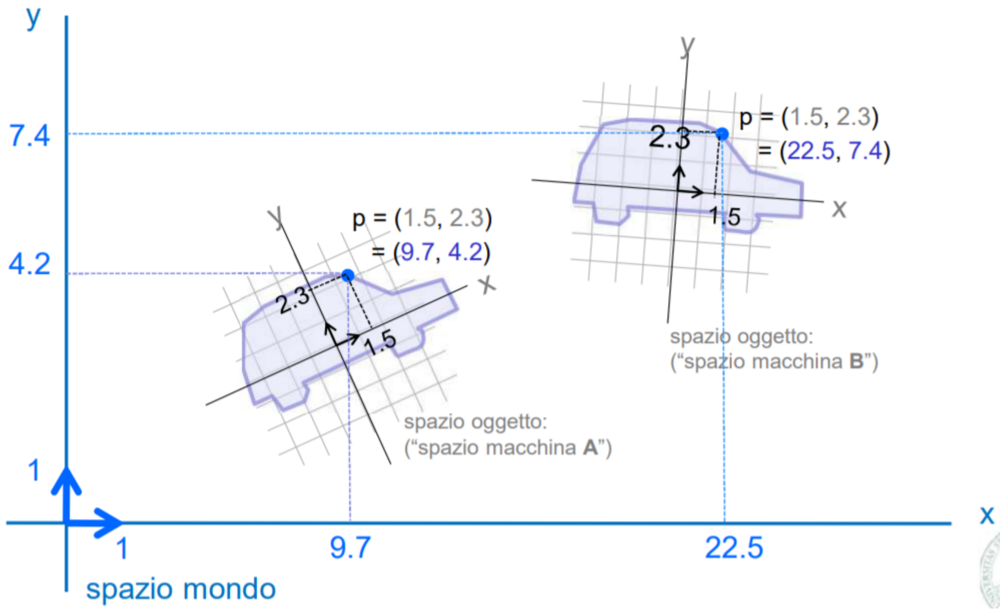
Trasformazione di proiezione

Trasforma lo **spazio vista** in **spazio clip**.
È determinata dai *parametri intrinseci* della camera (ad esempio la distanza focale, la posizione del POV etc.).

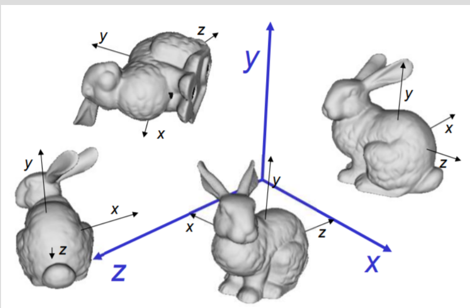
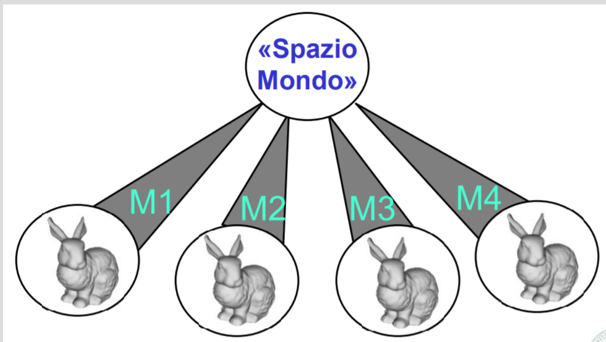
Le trasformazioni - descrizione

Modellazione

La trasformazione di modellazione deve portare ciascun oggetto 3D in spazio oggetto in un unico spazio 3D (la scena 3D) che è lo spazio mondo. Per farlo, applichiamo una matrice di modellazione a ciascun punto e vettore di ciascun oggetto, in modo da cambiare il sistema di riferimento di quell'oggetto da quello di spazio oggetto a quello di spazio mondo:

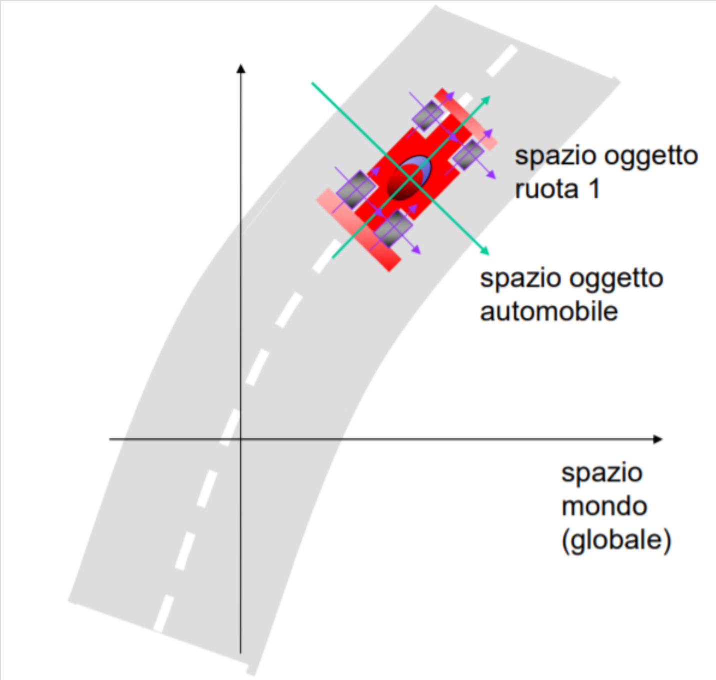


La matrice di modellazione di un oggetto determina quindi in che modo un oggetto è posizionato/dimensionato nella scena 3D. Dal momento che ogni istanza di un oggetto ha una propria matrice di modellazione, significa che possiamo rappresentare la scena come un grafo (***scene graph***), che avrà come radice lo spazio mondo, e come figli gli oggetti della scena:

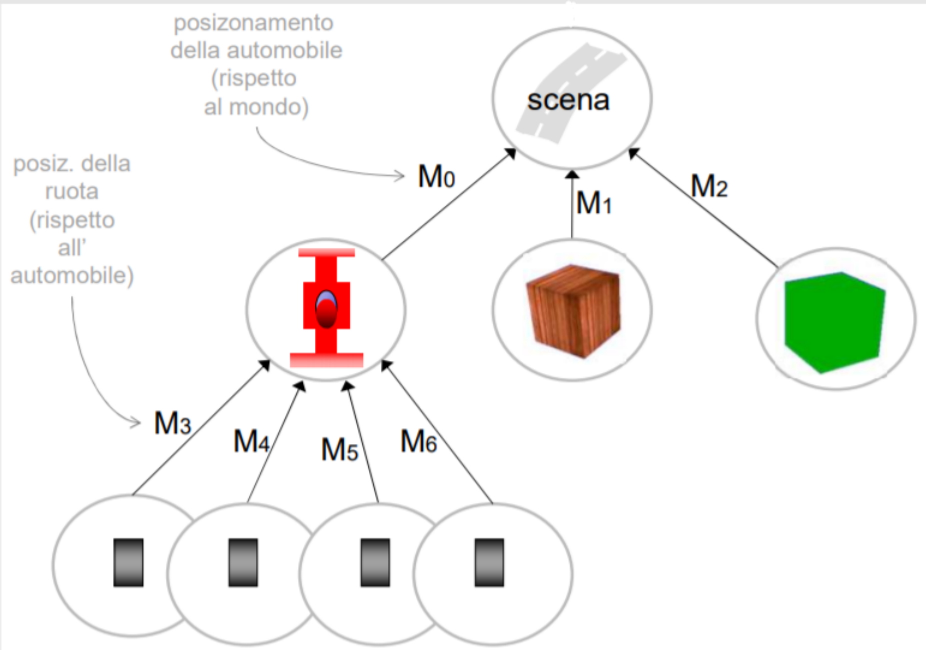
	
Scena 3D con 4 istanze dello stesso oggetto in posizioni diversa, quindi con M_M *diversa*	*La stessa scena rappresentata come scene graph: ogni istanza ha una* M_M *diversa e quindi è in un ramo diverso*

Il fatto di distinguere spazio oggetto da spazio mondo ha anche questo vantaggio in termini di memoria, in quanto un unico oggetto in memoria può essere istanziato in punti diversi della scena. Inoltre per modificare la posizione di un oggetto in scena, si tratta di cumulare una nuova M_M su quella vecchia, senza dover modificare le coordinate in spazio oggetto.

Lo scene graph può inoltre avere più livelli, in quanto possiamo rendere degli oggetti dipendenti dalle coordinate di altri oggetti, quindi avere idealmente dei "sottospazi mondo", che hanno comunque radice nell'unico spazio mondo. Ad esempio:



Le ruote della macchina sono portate in spazio oggetto della macchina, e sarà la macchina che sarà portata in spazio mondo.



Lo scene graph della scena mostra come le ruote siano dipendenti dalla spazio macchina, che a sua volta è dipendente dallo spazio mondo.

Per cumulare le diverse matrici di modellazione, si procede dalle foglie verso la radice. Questo significa che per calcolare la posizione r in spazio mondo di una ruota, basterà fare:

$$\overbrace{(M_0 \cdot \underbrace{(M_3 \cdot \underbrace{r}_{\text{coordinate spazio ruota}}))}_{\text{coordinate in spazio macchina}}}^{\text{coordinate spazio mondo}} = \overbrace{\underbrace{(M_0 \cdot M_3)}_{\text{matrice di modellazione della ruota}} \cdot r}^{\text{coordinate spazio mondo}}$$

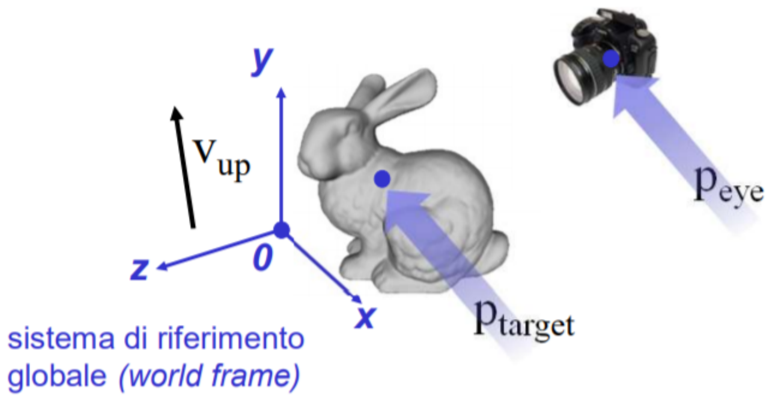
Questo ha un grosso vantaggio, cioè modificando la matrice di modellazione della macchina M_0, modifico anche la posizione delle ruote in spazio mondo (in quanto la matrice della ruota dipende anche da M_0), ma nel modo corretto (le ruote seguono la macchina).

Vista

La trasformazione di vista ha come scopo quello di descrivere le scena 3D secondo le coordinate della camera, o più in generale dell'osservatore. Cambiare la posizione dell'osservatore/camera cambia la matrice di vista, così come tutti gli altri *parametri estrinseci*, i quali nel dettaglio sono:

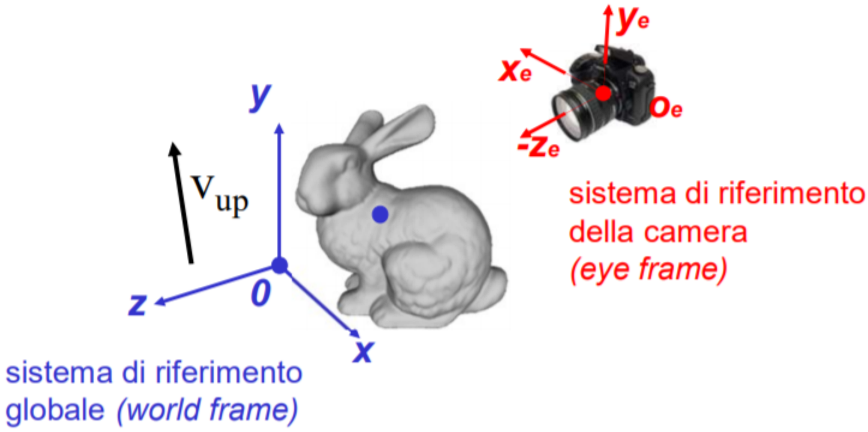
- posizione della camera
- posizione del target
- vettore alto (*up*)

Tutti questi parametri sono punti o vettori espressi in spazio mondo.



In generale avremo quindi il passaggio da spazio mondo allo spazio vista, le cui coordinate (più o meno standard) sono descritte sopra.

Idealmente, l'obiettivo della trasformazione di vista è quello di definire la scena 3D, non più "rispetto a sé stessa" (spazio mondo) ma rispetto alla camera/osservatore:



Per costruire la matrice di vista, costruiamo prima l'inversa, cioè la matrice M che va da spazio vista a spazio mondo, per poi invertirla.

Dichiariamo l'origine degli assi e i tre vettori di base, a cui dovranno essere assegnate coordinate espresse in spazio mondo (il nostro spazio di arrivo).

```
vec3 oe; //origine eye (camera)
vec3 xe, ye, ze; //vettori base di eye
```

L'origine `oe` è data in input come valore `p_eye`.

Ricaviamo le coordinate in spazio mondo dell'asse `ze` :

```
ze = p_eye - p_target /*l'asse z della camera in spazio mondo
coincide                          con il vettore che va dal
target al POV*/
ze = normalize(ze);
```

Ricaviamo il vettore `xe` come prodotto cross del vettore `vup`, cioè il vettore che indica "l'alto", e `ze`:

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

Infine otteniamo `ye` come prodotto cross tra `ze` e `xe`:

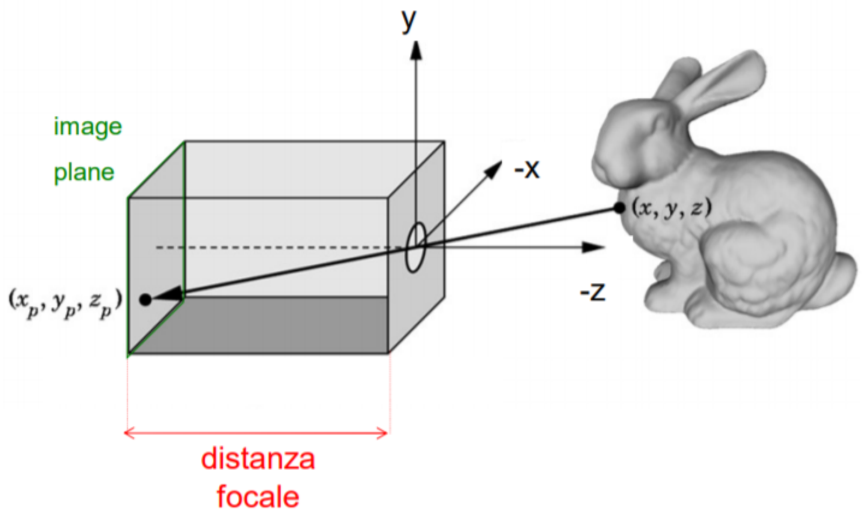
```
ye = cross(ze, xe); /*non serve normalizzare in quanto stiamo
facendo                          il cross tra vettori già
normalizzati*/
```

Invertendo la matrice M così ottenuta, otteniamo la matrice che trasforma lo spazio mondo in spazio vista, cioè la matrice che cerchiamo.

$$M_V = M^{-1} = \begin{pmatrix} x_e & y_e & z_e & o_e \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1}$$

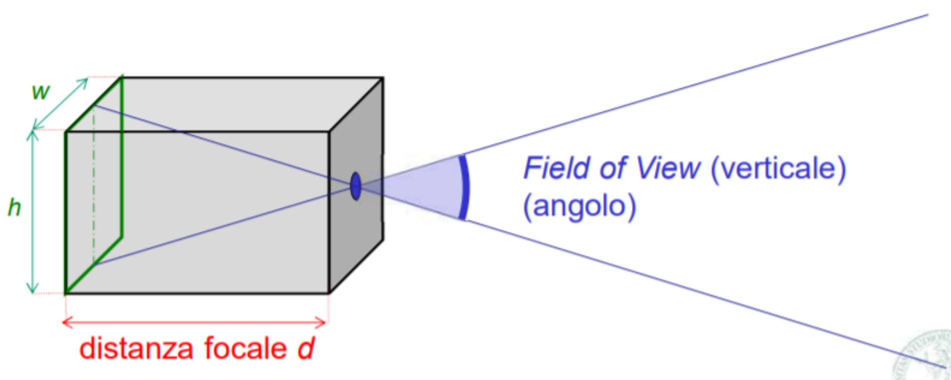
Proiezione

La trasformazione di proiezione deve portare un oggetto 3D in spazio vista in spazio clip, il quale lo possiamo considerare 2D come se fosse il piano immagine della camera, anche se sappiamo avere una terza dimensione necessaria per il depth test.



Per proiettare un punto in piano immagine, dobbiamo considerare i *parametri intrinseci* della camera:

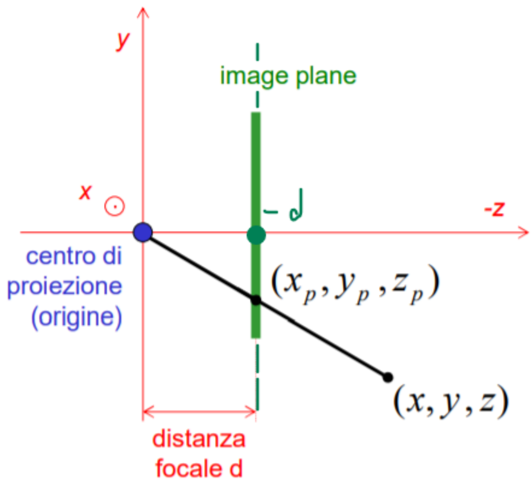
- dimensione piano immagine (w, h)
- distanza focale d oppure Field of View (da uno si può ricavare l'altro)



È importante notare che all'aumentare di d si riduce il FoV (effetto zoom), mentre al ridursi di d aumenta il FoV (effetto grandangolo).

Per comodità, infine poniamo il piano immagine di fronte al POV come abbiamo fatto nel ray tracing.
A questo punto possiamo considerare il problema di costruire la matrice di proiezione che dovrà portare i punti in spazio vista sul piano immagine.

Modellando il problema avremo la seguente situazione:



dove i tre assi sono quelli dello spazio clip e vogliamo proiettare il punto (x, y, z) appartenente allo spazio vista sul piano immagine. Si può osservare che:

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

infatti per porre il punto in spazio vista sul piano immagine, ciò che deve variare è la z del punto, quindi bisogna porre $z = -d$, in quanto $z_p = -d$ da come si nota in figura. Quindi avremo

$$kz = -d \Rightarrow k = -d/z$$

quindi:

$$\begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = k \begin{pmatrix} x \\ y \\ z \end{pmatrix} \text{con } k = -d/z$$
$$\iff \tag{1}$$
$$\begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = \begin{pmatrix} -dx/z \\ -dy/z \\ -dz/z \end{pmatrix} = \begin{pmatrix} -dx/z \\ -dy/z \\ -d \end{pmatrix}$$

Quest'ultimo risultato è giustificato geometricamente anche dalla prospettiva. Infatti all'aumentare di z, cioè quando gli oggetti si allontanano dalla camera, gli oggetti vengono scalati sulla x e sulla y di $1/z$, cioè vengono rimpiccioliti! Ovviamente al contrario z si ridurrà ingrandendo gli oggetti. Inoltre vi è anche il ruolo della distanza focale d, che come vedremo simulerà anche le proporzioni che gli oggetti mantengono rispetto al suo valore.

Quello che dovremmo fare ora è costruire la matrice di proiezione M_P tale che:

$$M_P \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} -dx/z \\ -dy/z \\ -d \\ 1 \end{pmatrix}$$

questa uguaglianza è impossibile, non si può mantenere 1 come coordinata omogenea (la trasformazione non è affine), ma bisogna usare la versione generalizzata delle coordinate omogenee:

$$M_P \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ -z/d \end{pmatrix} \tag{2}$$

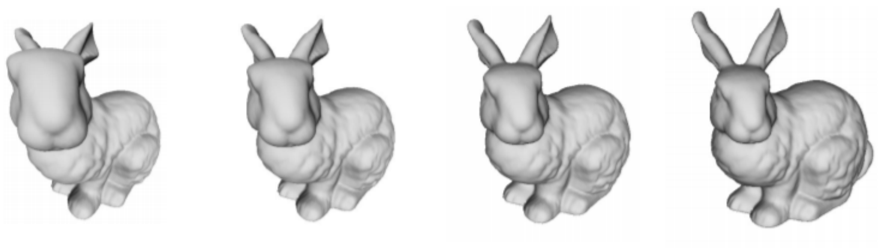
infatti dividendo per $w = -z/d$ ottengo la coordinate cartesiane trovate nel punto (1):

$$\underbrace{\begin{pmatrix} x \\ y \\ z \\ -z/d \end{pmatrix}}_{\text{coordinate omogenee}} \implies \underbrace{\begin{pmatrix} -dx/z \\ -dy/z \\ -d \end{pmatrix}}_{\text{coordinate cartesiane}}$$

Possiamo quindi trovare M_P che soddisfa la (2):

$$M_P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/d & 0 \end{pmatrix}$$

Da notare che per d piccolo avremo oggetti sproporzionati ma maggiore FoV (grandangolo), mentre per valori di d maggiori gli oggetti manterranno le proporzioni ma avremo un FoV ristretto (zoom):



M_P può essere riscritta anche in modo diverso ponendo $w = -z$:

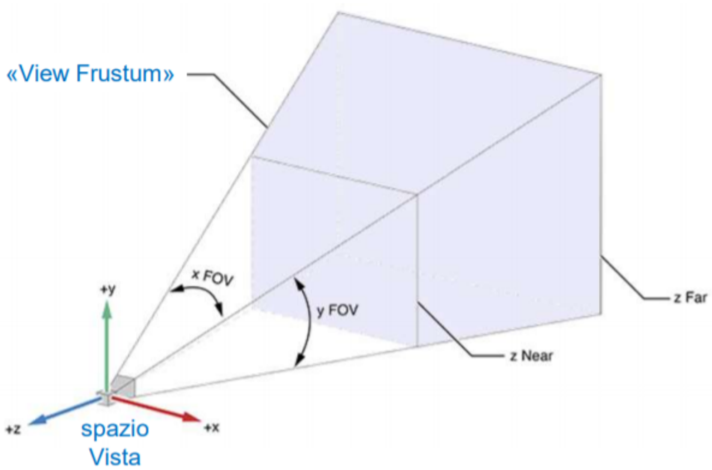
$$\underbrace{\begin{pmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & -1 & 0 \end{pmatrix}}_{M_P} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} dx \\ dy \\ dz \\ -z \end{pmatrix}$$

La M_P prodotta non tiene in considerazione il fatto che in spazio clip vi sia anche la coordinata z da calcolare, necessaria per il depth test, infatti le coordinate in spazio che clip che otteniamo sono, in entrambe lo forme di M_P, le seguenti:

$$\begin{pmatrix} -dx/z \\ -dy/z \\ -d \\ 1 \end{pmatrix}$$

dove $-d$ è un valore di z costante, mentre noi vorremmo che variasse in base alla "profondità" dello spazio clip.

Possiamo quindi estendere il modello di proiezione di spazio vista su spazio clip:



I valori z_{near} e z_{far} sono decisi in spazio vista e dovranno diventare i valori +1 e -1 in spazio clip. Si tratta quindi di portare l'intervallo $[-z_{near}, -z_{far}]$ in spazio vista nell'intervallo $[-1, +1]$ in spazio clip.
Quindi, se z in spazio vista è pari a $-z_{near}$, z in spazio clip deve essere pari a -1.
Se z in spazio vista è pari a $-z_{far}$, z in spazio clip deve essere pari a $+1$.
Dobbiamo modificare quindi M_P in modo tale che consideri questi due valori per proiettarli in spazio clip in modo corretto, quindi:

$$\underbrace{\begin{pmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & A & B \\ 0 & 0 & -1 & 0 \end{pmatrix}}_{M_P} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} dx \\ dy \\ Az + B \\ -z \end{pmatrix}}_{\text{coordinate in spazio clip}} \implies \underbrace{\begin{pmatrix} -dx/z \\ -dy/z \\ -A - B/z \\ 1 \end{pmatrix}}_{\text{coordinate in spazio clip}}$$

Per quanto detto in precedenza:

$$\begin{cases} -A + \frac{B}{z_{near}} = -1 & \text{poichè se } z_V = -z_{near} \text{ allora } z_C = -1 \\ -A + \frac{B}{z_{far}} = +1 & \text{poichè se } z_V = -z_{far} \text{ allora } z_C = +1 \end{cases}$$

Risolvendo il sistema:

$$A = \frac{z_{far} + z_{near}}{z_{far} - z_{near}} \qquad \text{e} \qquad B = \frac{2 \cdot z_{far} \cdot z_{near}}{z_{far} - z_{near}}$$

Un'altra modifica da fare alla matrice di proiezione è quella di correggere le distorsioni dovute all'*aspect ratio* dello schermo cioè:

> L'**aspect ratio** è il rapporto tra la larghezza e l'altezza dello schermo, espressi in termini di risoluzione.
>
> $$\text{aspect ratio } a = \frac{resX}{resY}$$

L'aspect ratio viene considerato nel passaggio da spazio clip a spazio schermo, e un rapporto troppo elevato può creare distorsioni nell'immagine. Per correggere, modifichiamo un'ultima volta M_P:

$$M_P = \begin{pmatrix} d & 0 & 0 & 0 \\ 0 & d \cdot a & 0 & 0 \\ 0 & 0 & A & B \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

La M_P finale sarà dunque:

$$M_P = \begin{pmatrix} d & 0 & 0 & 0 \\ 0 & d \cdot a & 0 & 0 \\ 0 & 0 & \frac{z_f + z_n}{z_f - z_n} & \frac{2 \cdot z_f \cdot z_n}{z_f - z_n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Conclusione

Alla fine dell'applicazione delle matrici di modellazione, vista e infine proiezione, abbiamo finalmente i punti del modello 3D in spazio clip. La fase successiva del pipeline di rendering rasterization based è la rasterizzazione.