Classi utili di trasformazioni

- Isometrie (rototraslazione)
  - "Mantengono la magnitudine"
  - Rotaz + Traslaz
- Similitudini (trasformaz. *conformali*)
  - "Mantengono gli angoli"
  - Rotaz + Traslaz + Scaling uniforme
- Lineari (trasformaz. affini)

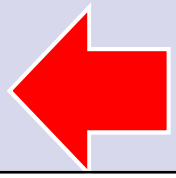
$$f(\alpha v_0 + \beta v_1) = \alpha f(v_0) + \beta f(v_1)$$

Nota:
sono chiuse rispetto a combinazione

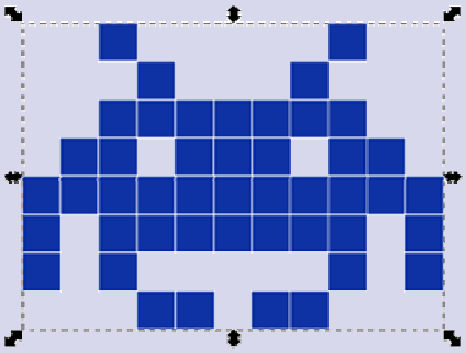
Classi utili di trasformazioni

- Isometrie (rototraslazione)
  - "Mantengono la magnitudine"
  - Rotaz + Traslaz
- Similitudini (trasformaz. *conformali*)
  - "Mantengono gli angoli"
  - Rotaz + Traslaz + Scaling uniforme
- Lineari (trasformaz. affini)

$$f(\alpha v_0 + \beta v_1) = \alpha f(v_0) + \beta f(v_1)$$



Demo in 2D



Passaggio preliminare:
Coordinate omogenee

Vettori posizione

Vettori direzione

$$\mathbf{p} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ 1 \end{bmatrix} \quad 1 \rightarrow \qquad \mathbf{a} = \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ 0 \end{bmatrix} \quad 0 \rightarrow$$

La 4ta cordinata “omogenea”

La coordinata w

Trasformazioni Affini

- Si possono esprimere come moltiplicazione con matrice

$$f\begin{pmatrix}\gamma_1\\ \gamma_2\\ \gamma_3\\ 1\end{pmatrix}=\begin{bmatrix}\alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14}\\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24}\\ \alpha_{31} & \alpha_{32} & \alpha_{33} & \alpha_{34}\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\gamma_1\\ \gamma_2\\ \gamma_3\\ 1\end{bmatrix}=\begin{pmatrix}\delta_1\\ \delta_2\\ \delta_3\\ 1\end{pmatrix}$$

sempre

coordinate affini punto di partenza

coordinate affini punto di arrivo

Trasformazioni Affini

- Caso vettori "direzione"

conta solo questo

$$f\begin{pmatrix}\gamma_1\\ \gamma_2\\ \gamma_3\\ 0\end{pmatrix}=\begin{bmatrix}\alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14}\\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24}\\ \alpha_{31} & \alpha_{32} & \alpha_{33} & \alpha_{34}\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\gamma_1\\ \gamma_2\\ \gamma_3\\ 0\end{bmatrix}=\begin{pmatrix}\delta_1\\ \delta_2\\ \delta_3\\ 0\end{pmatrix}$$

sempre

coordinate affini vettore di partenza

coordinate affini vettore di arrivo

Esempio: trasformazione di traslazione rigida

$$f\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{bmatrix} x+\alpha_x \\ y+\alpha_y \\ z+\alpha_z \\ 1 \end{bmatrix}$$

e cioè:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} + \begin{bmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \\ 0 \end{bmatrix}$$

vettore di traslazione

posso riscriverla come:

$$f\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & \alpha_x \\ 0 & 1 & 0 & \alpha_y \\ 0 & 0 & 1 & \alpha_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Trasformazione di Traslazione rigida

matrice di traslazione:

$$\mathbf{T}(\alpha_x,\alpha_y,\alpha_z) = \begin{bmatrix} 1 & 0 & 0 & \alpha_x \\ 0 & 1 & 0 & \alpha_y \\ 0 & 0 & 1 & \alpha_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

l'inversa è ovviamente:

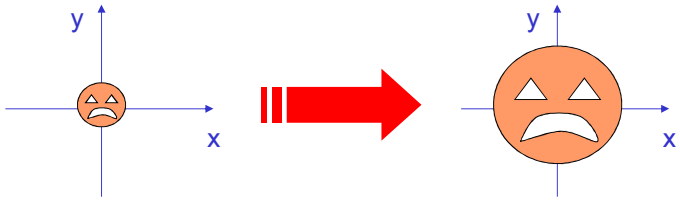
$$\mathbf{T}^{-1}(\alpha_x,\alpha_y,\alpha_z) = \mathbf{T}(-\alpha_x,-\alpha_y,-\alpha_z) = \begin{bmatrix} 1 & 0 & 0 & -\alpha_x \\ 0 & 1 & 0 & -\alpha_y \\ 0 & 0 & 1 & -\alpha_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Trasformazione di Traslazione rigida

cosa succede se la applico ad un *vettore*?

$$f\begin{pmatrix} x \\ y \\ z \\ \mathbf{0} \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & \alpha_x \\ 0 & 1 & 0 & \alpha_y \\ 0 & 0 & 1 & \alpha_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \mathbf{0} \end{bmatrix} = \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix}$$

Trasformazione di Scalatura uniforme

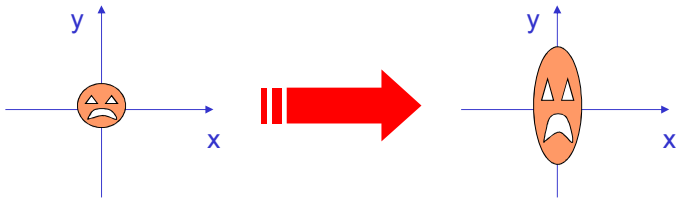


$$f\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{bmatrix} \gamma x \\ \gamma y \\ \gamma z \\ 1 \end{bmatrix} \qquad f\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{bmatrix} \gamma & 0 & 0 & 0 \\ 0 & \gamma & 0 & 0 \\ 0 & 0 & \gamma & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

matrice di scaling

$\mathbf{S}(\gamma)$

Trasformazione di Scalatura generica



$$f\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{bmatrix} \gamma_x x \\ \gamma_y y \\ \gamma_z z \\ 1 \end{bmatrix} \qquad f\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{bmatrix} \gamma_x & 0 & 0 & 0 \\ 0 & \gamma_y & 0 & 0 \\ 0 & 0 & \gamma_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

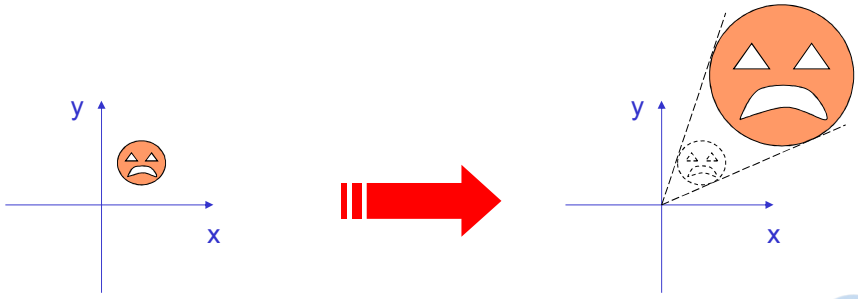
matrice di scaling

$$\mathbf{S}(\gamma_x, \gamma_y, \gamma_z)$$

inversa?

Trasformazione di Scalatura generica

nota: la scalatura applicata ai punti
"scala" anche la distanza dall'origine



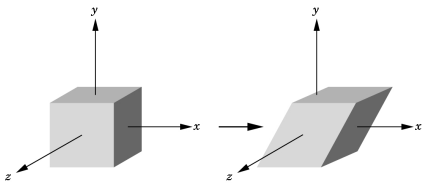
Trasformazione di Scalatura generica

- Osservazioni:
  - Fattori di scala inferiori a 1 avvicinano l'oggetto al punto fisso di riferimento (origine)
  - Fattori di scala maggiori di 1 lo allontanano
  - Se $s_x \neq s_y$ o $s_y \neq s_z$ le proporzioni dell'oggetto non sono mantenute (scalatura *non uniforme,* o *anisotropica*)
  - Se $s_x = s_y = s_z$ le proporzioni sono mantenute e si ha una *scalatura uniforme* (o *isotropica*)

Shearing

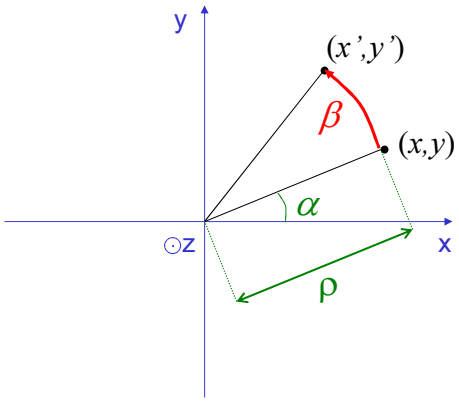
- Lo spostamento proporzionale alla pos *y*

$$x' = x + y\cot\theta$$
$$y' = y$$
$$z' = z$$



$$H_{xy}(\theta) = \begin{bmatrix} 1 & \cot\theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotazione attorno all'asse z



partenza:

$$x = \rho\cos\alpha$$
$$y = \rho\sin\alpha$$

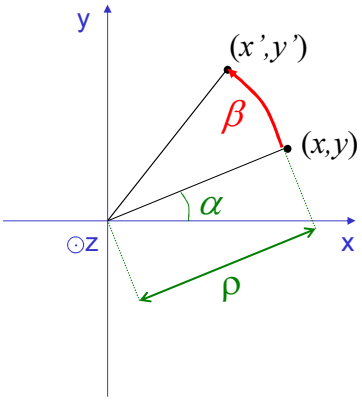
arrivo:

$$x' = \rho\cos(\alpha+\beta) = \rho\cos\alpha\cos\beta - \rho\sin\alpha\sin\beta = x\cos\beta - y\sin\beta$$
$$y' = \rho\sin(\alpha+\beta) = \rho\cos\alpha\sin\beta + \rho\sin\alpha\cos\beta = x\sin\beta + y\cos\beta$$

Rotazione attorno all'asse z

$$x' = x\cos\beta - y\sin\beta$$
$$y' = x\sin\beta + y\cos\beta$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = R_Z(\beta)\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x\cos\beta - y\sin\beta \\ x\sin\beta + y\cos\beta \\ z \\ 1 \end{bmatrix}$$



$$R_Z(\theta) = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & 0 \\ \sin\beta & \cos\beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotazione attorno all'asse x, y, o z

$$R_X(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_Y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

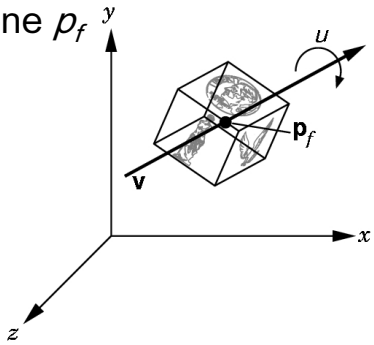
$$R_Z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

e le inverse?

$$R_X(\theta)^{-1} = R_X(-\theta) = R_X(\theta)^{\mathrm{T}}$$

Rotazioni generiche

- Una rotazione generica è definita da:
  - angolo *u*,
  - asse *v*
  - punto di applicazione p_f
- come si fa?

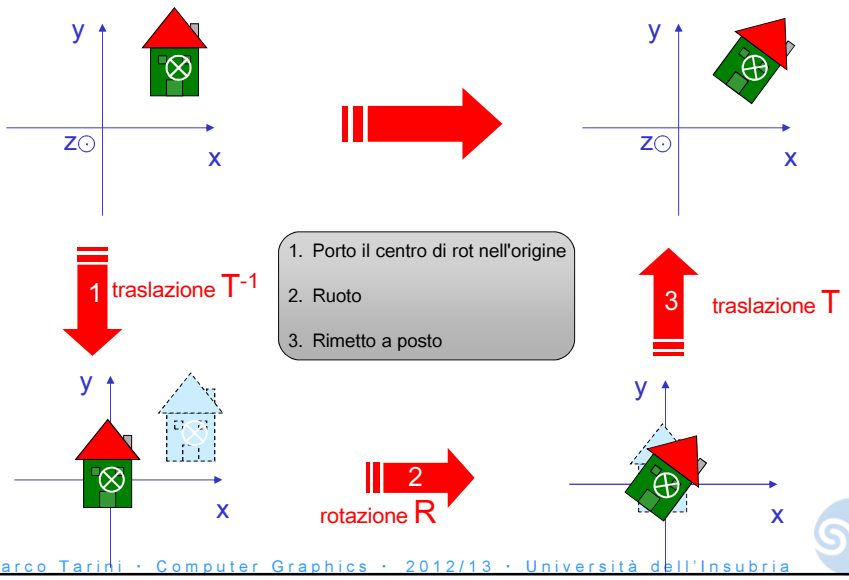


da XKCD

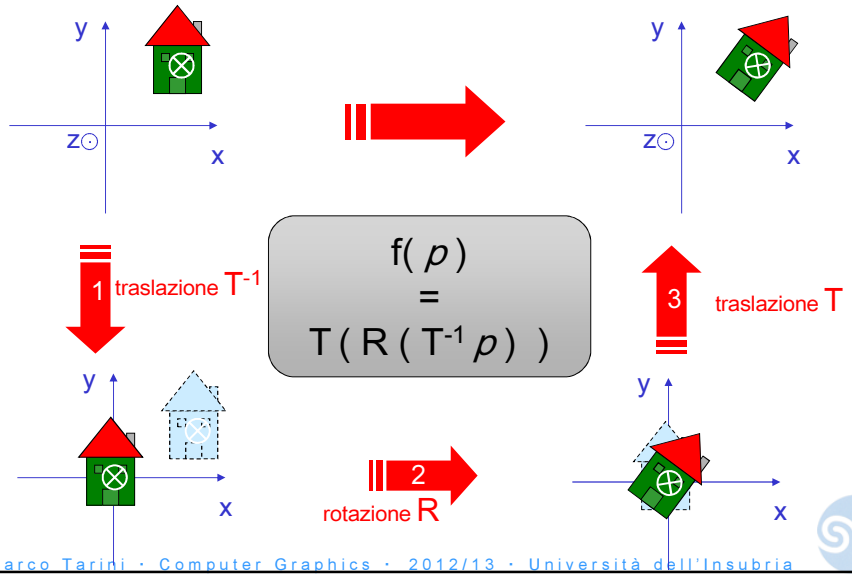
http://xkcd.com/184/

$$\begin{bmatrix} \cos 90^\circ & \sin 90^\circ \\ -\sin 90^\circ & \cos 90^\circ \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} a_2 \\ -a_1 \end{bmatrix}$$

Rotazione intorno ad un asse *parallelo* all'asse z



Rotazione intorno ad un asse *parallelo* all'asse z



Composizione di trasformazioni

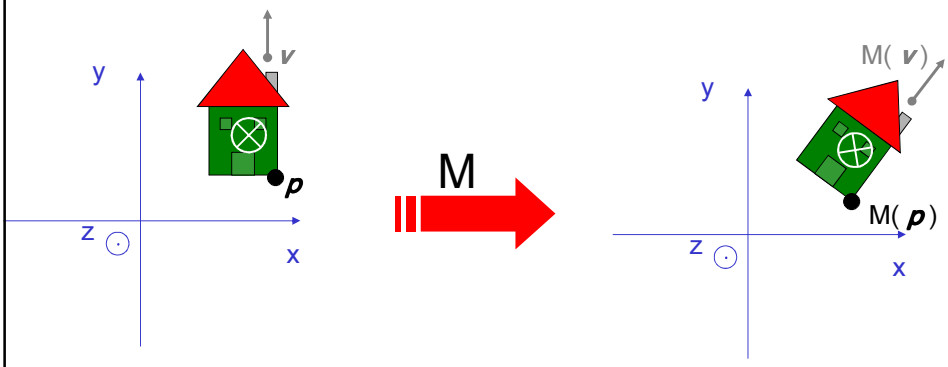
- Moltiplicazione matrici (vettori) ha la propretà associativa

$$f(p) = T (R (T^{-1} p)) = (T R T^{-1}) p$$

una matrice M 4x4 che fa tutto.

- considerazioni sull'efficienza
- cosa possiamo dire sulla forma di M ?
- cosa succede se moltiplichiamo un *vettore* per M ?

Punti VS vettori



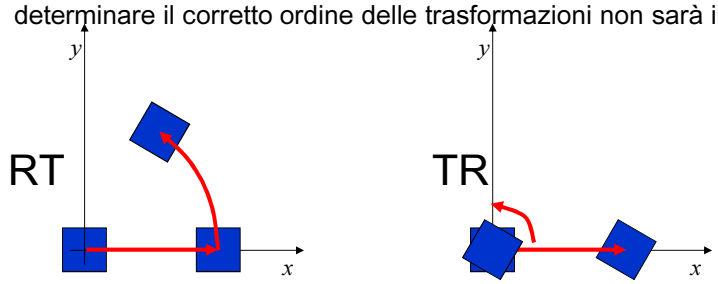
p **= (* , * , * , 1)** punto all'angolo della casa **(punto)**
v **= (* , * , * , 0)** velocità vettoriale del fumo **(vettore)**

Ripassino

- Attenzione all'inversione: $(AB)^{-1} = B^{-1}A^{-1}$
- Associativa si, ma commutativa no!

$$AB \neq BA$$

- previsione:
  determinare il corretto ordine delle trasformazioni non sarà intuitivo



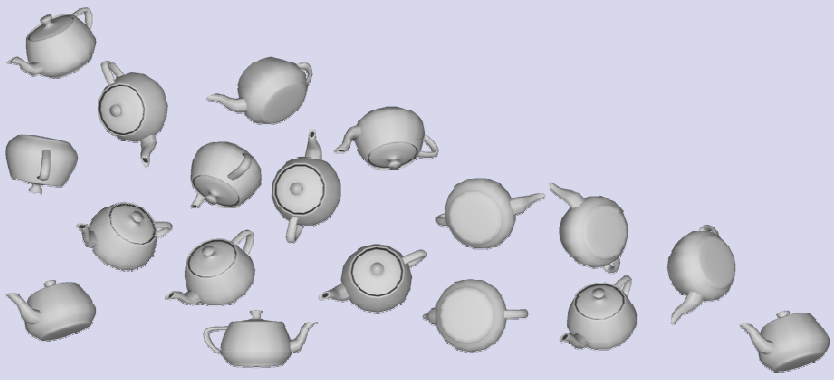
Classi utili di trasformazioni

- Isometrie (rototraslazione)
  - "Mantengono la magnitudine"
  - Rotaz + Traslaz
- Similitudini (trasformaz. conformali)
  - "Mantengono gli angoli"
  - Rotaz + Traslaz + Scaling uniforme
- Lineari (trasformaz. affini)

$$f(\alpha v_0 + \beta v_1) = \alpha f(v_0) + \beta f(v_1)$$

Come rappresento le rotazioni in 3D?

- Cioè anche gli orientamenti di un oggetto nello spazio

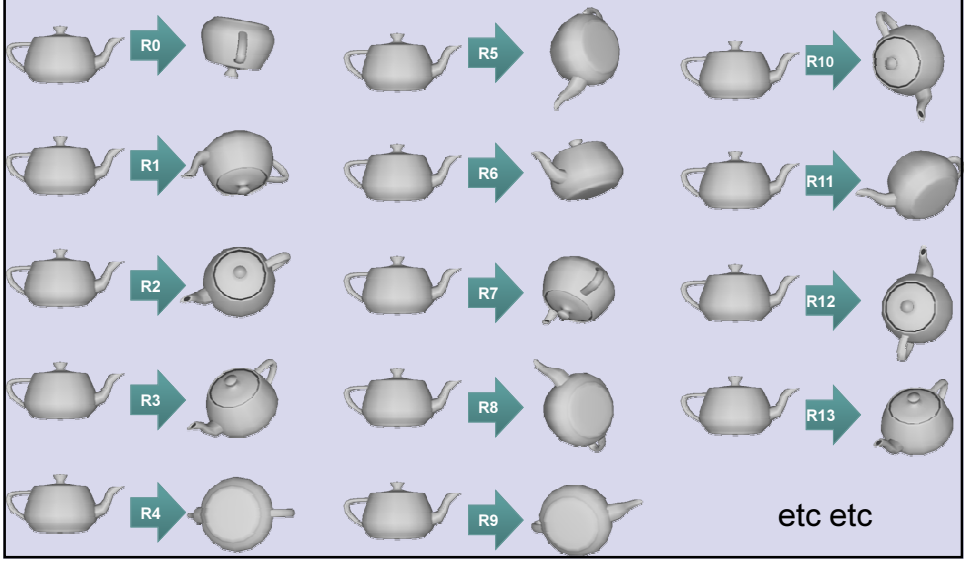


Reminder

- Tutte e sole le isometrie (trasf. rigide)
  = roto-traslazioni
  = rotazioni (*) + traslazioni
- Rotazioni (*) :
  - quante possibili?
  - come rappresentarle (internamente)?

(*) generiche = attorno ad assi passanti per l'origine

Rotazioni in R3: quante possibili?



Rotazioni in R3: quante possibili?

(e ovviamente includono l'identità)

Reppresentazioni possibili per rotazioni

- Buone (o meno) per:
  - compattezza
    - quanto sono prolisse in memoria?
  - facilità di applicazione
    - quanto è oneroso applicare ad uno (o ventimila) punti / vettori?
  - interpolabilità
    - è possibile/facile trovare un'inerpolazione fra N rotazioni date?
    - quanto è "buono" il risultato
  - combinabilità
    - è facile trovare la risultante di N rotazioni date, eseguite in successione?
  - invertibilità
    - è facile trovare l'inversa?

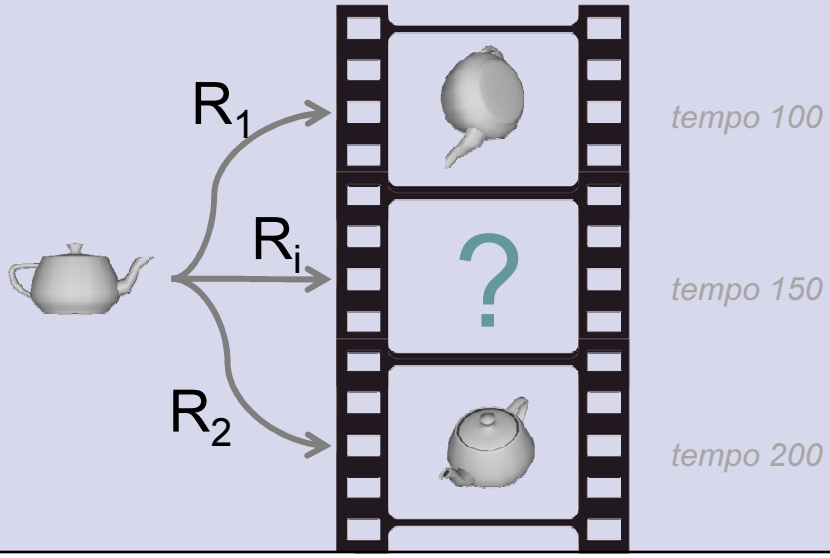
Per paragone: reppresentazione delle *traslazioni*

- Banale:
  vettore di displacement (tre float)!
  - perfetta secondo tutti i criteri (verificare!)

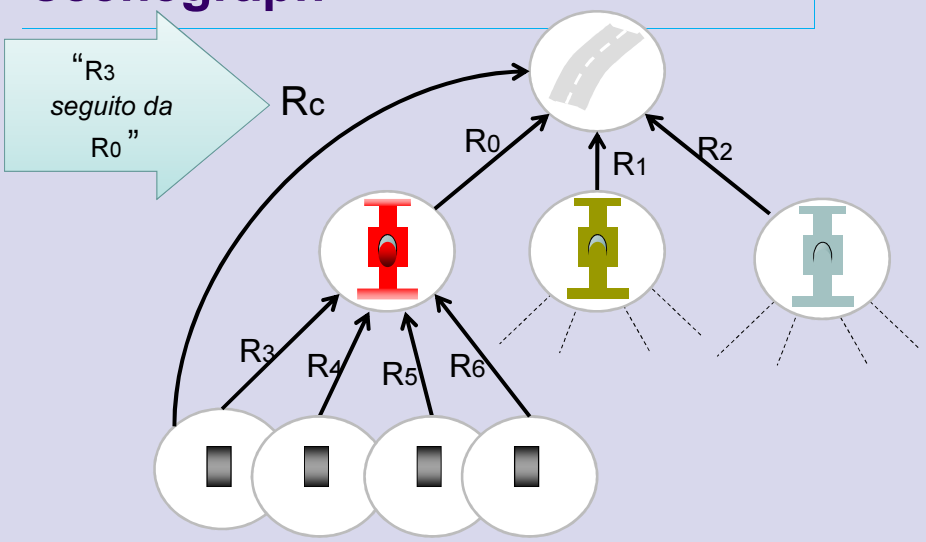
Reppresentazioni per rotazioni

- Molte possibili,
  con vantaggi e svantaggi diversi
- Tutte molto diffuse ed usate
- Modi per passare da una rappr. all'altra?

Perché è utile *interpolare* rotazioni: esempio: animazioni



Perché è utile *cumulare* rotazioni: scenegraph



Reppresentazioni principali delle rotazioni

- Matrici 3x3
- Angoli di Eulero
- Asse + angolo
- Quaternioni

Modo 1: matrice 3x3 (9 floats)

- (sottomatrice 3x3 della matrice di rot 4x4)

$$\begin{bmatrix} R & \begin{matrix}0\\0\\0\end{matrix} \\ \begin{matrix}0 & 0 & 0\end{matrix} & 1 \end{bmatrix}$$

- come sappiamo, R ortonormale con det = 1

Modo 1: matrice 3x3 (9 floats)

- Prolissa (9 numeri invece di 3)
- Abb. facile da applicare (molt matrice-vettore)
  - come sappiamo, cumulabile con qualunque altra trasf. affine
- Abb. facile da cumulare (molt matrice-matrice)
- Facilissima da invertire (trasposiz matrice)
- Problematica da interpolare:

perché?

$$k\,R_0 + (1-k)\,R_1 = M$$

in genere NON di rotazione (non ortonormale)

Reppresentazioni principali delle rotazioni

- Matrici 3x3
- Angoli di Eulero
- Asse + angolo
- Quaternioni

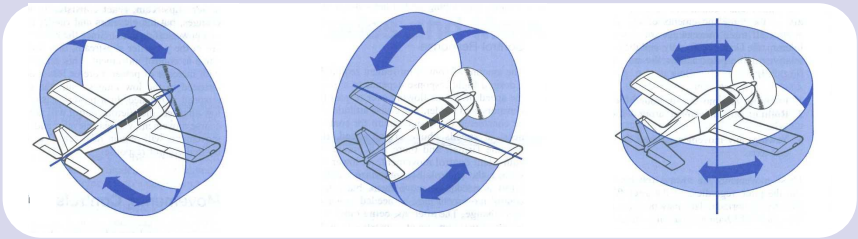
Modo 2: angoli di eulero (3 floats)

- Qualunque rotazione (*)
  può essere espressa come:
  - rotazione lungo asse X (di α gradi), seguita da:
  - rotazione lungo asse Y (di β gradi), seguita da:
  - rotazione lungo asse Z (di γ gradi):
- Angoli α β γ :
  "angoli di Eulero" di quella rotazione
  - (quindi: le "coordinate" di quella rotaz)

oridine (X-Y-Z) arbitrariamente scelto, (ma 1 volta x tutte)

Modo 2: angoli di eulero (3 floats)

- In linguaggio nautico / areonautico:
  angoli di "rollio, beccheggio, imbardata"



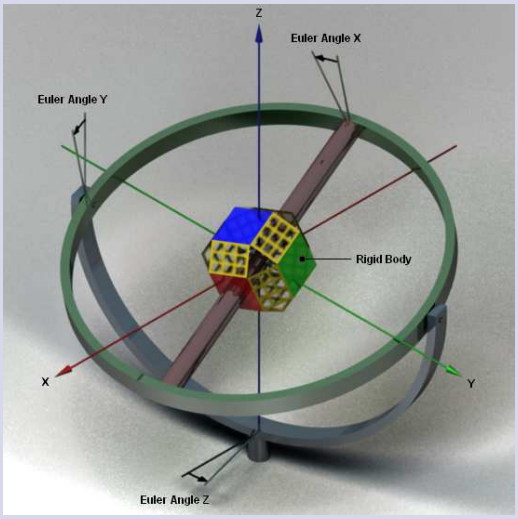
rollio (*roll*)

beccheggio (*pitch*)

imbardata (*yaw*)

Modo 2: angoli di eulero (3 floats)

- Implementaz. fisica: "mappamondo a tre assi"



Modo 2: angoli di eulero (3 floats)

- Compattezza: perfect!
- Da applicare: facilino
  - (se a molti punti, meglio passare a matrice)
- Da interpolare: possibile…
  - intrerpolaz dei tre angoli
  - (occhio ad interpolare angoli: ricordarsi equivalenza angoli: α = α +360 (k))
- …ma risultati non sempre intuitivi)
- Da cumulare e invertire: problematico…

perché sommare e invertire gli angoli non funziona?

da: angoli di eulero
a: matrice 3x3

- Facile!

$$M = R_z(\gamma) \cdot R_y(\beta) \cdot R_x(\alpha)$$

- Il viceversa?
  - (solo a suon di conti e funz trigon. inverse)

Reppresentazioni principali delle rotazioni

- Matrici 3x3
- Angoli di Eulero
- Asse + angolo
- Quaternioni

Modo 3: asse e angolo

- Qualunque rotazione (*) data può essere espressa come:
  - una (sola!) rotazione (di un **angolo**) attorno ad un **asse**

opportunamente scelti

- **Angolo:** uno scalare (1 float)
- **Asse:** un vettore normale (3 float)
  - (asse passante per l'origine)

Modo 3: asse e angolo

- Compattezza: molto buono
- Interpolazione: ottimo!
  - interpolare asse, intrerpolare angolo (nb: rinormalizzare asse!)
- Applicare: male ☹
  - modo migliore: passare a matrice 3x3 (o a quaternione)
- Cumulare: male ☹
- Invertire: facilissimo
  - (invertire angolo *oppure* asse)

Modo 3: asse e angolo: variante

- asse: v (vett normale, $|v| = 1$)
- angolo: α (scalare)
- rappresentarli internamente come 1 solo vett: v' (3 float in tutto)
  $v' = \alpha\, v$
  - angolo α = $|v'|$
  - asse v = $v' \,/\, |v'|$
  - (nota: se angolo = 0, asse si perde… infatti non conta)
- Più coinciso, ma per il resto equivalente
- (comune es. in fisica)

Reppresentazioni principali delle rotazioni

- Matrici 3x3
- Angoli di Eulero
- Asse + angolo
- Quaternioni

Modo 4: “quaternioni” (4 float)

- Solo alcuni cenni:
  - analogo (in 4D) dei numeri complessi (in 2D)
  - struttura simile ad asse + angolo:
    $q = (asse_x, asse_y, asse_z, \cos(angolo / 2))$
    $|q| = 1$
  - teoria molto elegante e solida
  - Ecco un altro “vec4” molto utile!
  - storia:
    - roba di mezz’800!
    - cross e dot products emergono dalla loro teoria (!)
    - nati proprio per questo scopo (rappresentare rotazioni)

Modo 4: “quaternioni” (4 float)

- Compattezza: abbastanza bene
  - 4 floats
- Cumulare: facillimo ;)
  - molt. di quaternioni
- Invertire: facillimo ;)
  - conigazione di quaternioni
- Interpolare: facillimo ;) e best results!
  - Interpolaz di quaternioni,
- Applicazione diretta: facillimo ;)

Rotazioni in unity

- Nella GUI del game tools:
  - Euler Angles
- Internamente:
  - Quaternions
  - Dunque, negli scipts, class quaternion

Trasformazioni in unity

- Lineari (trasformaz. affini)

$$f(\alpha v_0 + \beta v_1) = \alpha f(v_0) + \beta f(v_1)$$

- Similitudini (trasformaz. conformali)
  - "Mantengono gli angoli"
  - Rotaz + Traslaz + Scaling ~~uniforme~~ anisotropico
- Isometrie (rototraslazione)
  - "Mantengono la magnitudine"
  - Rotaz + Traslaz