Grafica Computazionale

Lab 2. Virtual Trackball

Fabio Ganovelli
fabio.ganovelli@gmail.com
a.a. 2006-2007

Vedere la scena

- Qualunque applicazione fornisce qualche modalità di cambiare la vista della scena
- Ciò può avvenire in 2 modi:
  - Si sposta la scena (es: la scena ruota)
  - Si sposta l'osservatore (es: noi giriamo intorno alla scena)
- Se la vista prodotta è la stessa al matrice di modello è la stessa in entrambi i casi. Cambia il paradigma
  - In un CAD tipicamente si trasla/ruota/scala la scena
  - In un First Person Shooter si trasla/ruota il punto di vista

Rotazione della scena

- La capacità di ruotare la scena è un mattone fondamentale dell'interfaccia
- Quanti modi abbiamo di farlo?
  - Specificare gli Euler Angles (pitch, roll,yaw)
  - Specificare un asse e un angolo di rotazione
  - Usare i quaternions
- In ogni caso si ottiene una matrice di rotazione da applicare

Euler angles

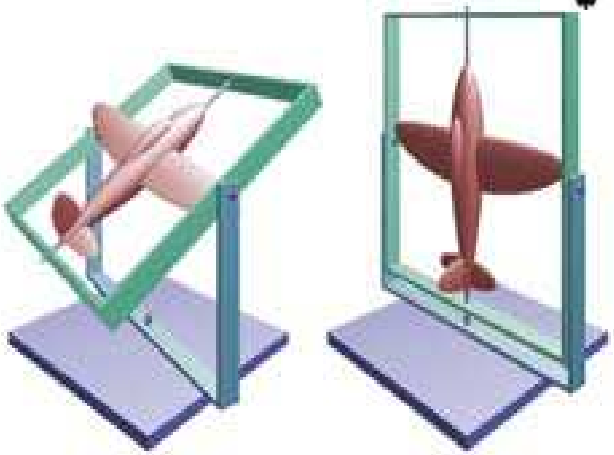
- Il più facile da implementare. Si specificano gli angoli di pitch, roll e yaw
  - Associando un evento all'incremento/decremento di ogni angolo


Pitch

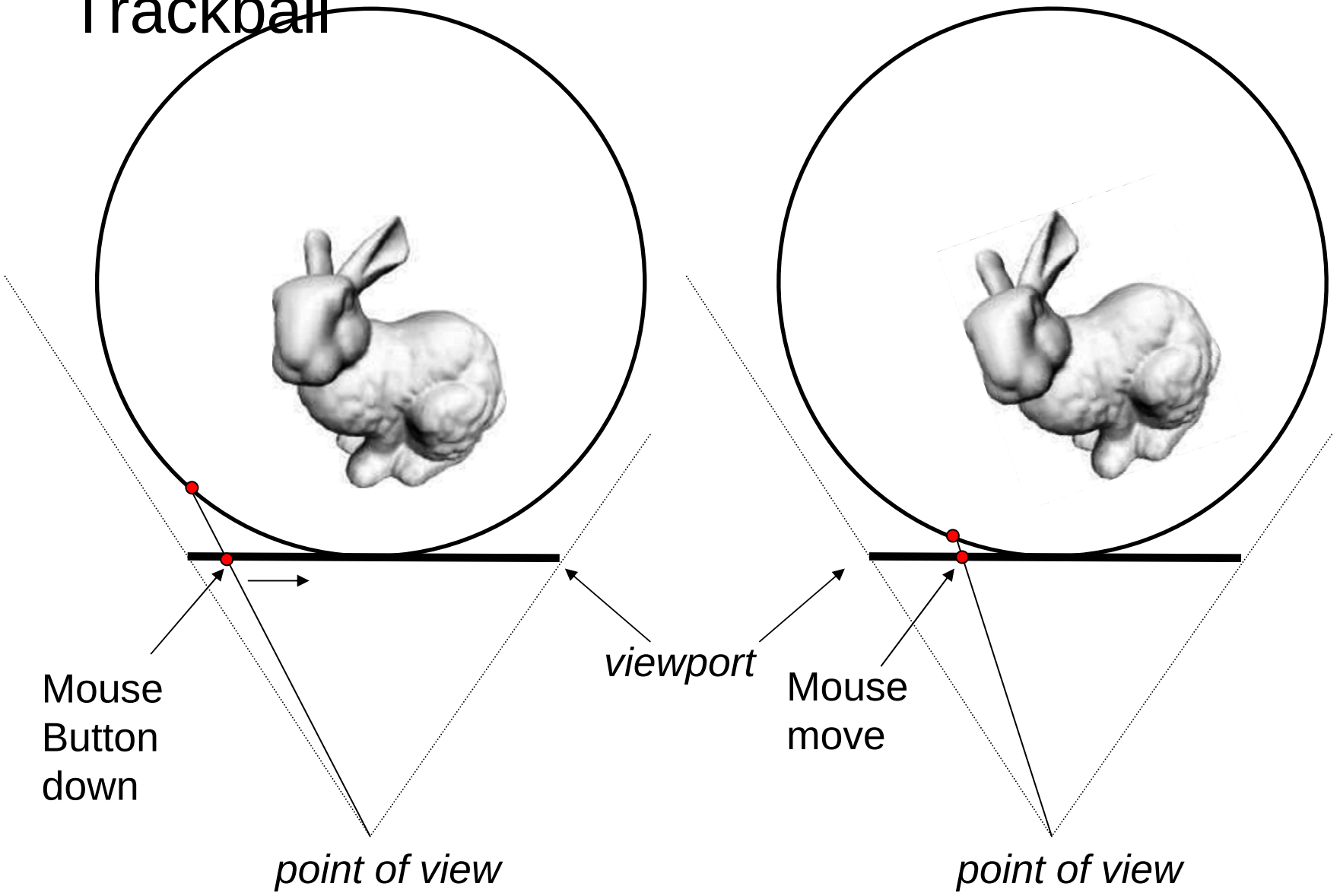

Roll


Yaw

- Problema Gimbal Lock
  - In alcune situazioni le rotazioni fatte su un asse possono annullare su un altro asse
  - Se il pitch è a 90° yaw e roll tendono a vicenda.



Trackball


Mouse
Button
down
viewport
Mouse
move
point of view
point of view

Da viewport a window coordinates

- Il movimento del mouse viene passato in device coordinates, cioè nelle coordinate della finestra di sdl (o equivalente)
- A noi però servono in object space
  - Richiamo: le trasformazioni di OpenGL
    - Modelview matrix
      - Model space TO object space
      - Object space TO view space
    - Projection Matrix
      - View space TO canonical view volume [-1,1]x[-1,1]x[-1,1]
    - Viewport Matrix
      - Window (NDC) TO viewport

Da viewport a window coordinates

- Occhio alla “z”
  - Va da -1 a +1 nel volume di vista canonico
  - Va da “min” a “max” nel depth buffer (min=0 e max = 1 di default in OpenGL
- I passi
  - Riportare z nel volume di vista canonico (z’ = z*2-1)
  - Applicare l’inversa della projection matrix
  - Applicare l’inversa della modelview matrix
- Dobbiamo farli a mano?

gluUnProject(..)

- Per fortuna abbiamo gluUnProject!
  - **int gluUnProject(**

```
GLdouble  winX,
GLdouble  winY,
GLdouble  winZ,
const GLdouble * model,
const GLdouble * proj,
const GLint * view,
GLdouble*  objX,
GLdouble*  objY,
GLdouble*  objZ)
```

Punto in window coordinates

Modelview matrix

Projection matrix

viewport

risultato

Recuperare le matrici

- glGetDoublev(GL_MODELVIEW_MATRIX,double *mm)
  - con double mm[16];
- glGetDoublev(GL_PROJECTION_MATRIX,double *pm)
  - con double pm[16]
- glGetIntegerv(GL_VIEWPORT, int *vp)
  - con int vp[4]

- Domanda: come mai per la viewport si non si prende la matrice?