Corso di *Tecniche Avanzate per la Grafica*

OpenGL & GLSL

Docente:

Massimiliano Corsini

Laurea Specialistica in Informatica

Facoltà di Scienze MM. FF. NN.
Università di Ferrara

Introduzione

- Open Graphics Language
  - Libreria C
  - Cross-platform
  - Qualche centinaio di routines
  - Specifiche definite dall'***ARB***
  - Sito di riferimento: ***www.opengl.org***

Breve Storia

- Inizialmente sviluppato da Silicon Graphics

- Ora:
  OpenGL **A**rchitecture **R**eview **B**oard
  - mantiene e aggiorna le *specifiche*
  - versione attuale: **2.0**
  - una compagnia, un voto

- Ci sono anche le *estensioni* private
  - Soprattutto ATi e NVIDIA.

GLU e GLUT

- OpenGL è il layer di base
- GLU (GL Utilites)
  - insieme di funzioni di utility costruite sopra OpenGL, piu'comode da usare
  - esempio

```
void gluLookAt(eyex,eyey,eyez,
               cx,cy,cz,
               upx, upy, upz);
```

- GLUT e' il Toolkit di interfaccia con il SO
- Wgl e GLx sono i sottoinsiemi di OpenGL che dipendono dal SO

Sintassi

- Tutte le funzioni di OpenGL hanno la sintassi tipo:
  - glFunctionXXX
- XXX specifica il tipo dei parametri
  - Esempio:
    ```
    glColor3f(float, float, float);
    glColor3fv(float*);
    ```
    f: float d: double v: vettore

    b: byte I: integer ecc…
  - Non e' C++ …

Macchina a stati

- L'OpenGL funziona come una ***macchina a stati***:
  - colore corrente
  - posizione luci
  - Matrici di trasformazione

  fanno parte dello stato corrente
- Molti comandi OpenGL modificano soltanto lo stato corrente

Matrici in OpenGL

Sono memorizzate per colonne => ***column-major order***

$$\begin{bmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \end{bmatrix}$$

Matrici e stato

- Lo stato comprende due matrici (e due stacks)
  - Matrice *Model-View*
  - Matrice di *Proiezione*
- Una di queste è sempre la matrice di lavoro (matrice corrente sulla quale avvengono le operazioni)
- Per cambiare la matrice di lavoro si utilizza
  - `glMatrixMode(GL_MODELVIEW);`
  - `glMatrixMode(GL_PROJECTION);`

Matrici e stato

- Per rimpiazzare la matrice di lavoro
  - `glLoadIdentity();`
  - `glLoadMatrixf(float *m);`
- Tutti i comandi che operano sulla matrice di lavoro la modificano moltiplicandola per un’altra matrice

Trasformazioni Geometriche

- Rotazioni
  - `glRotatef(angle, ax, ay, az);`
- Traslazioni
  - `glTranslatef(tx,ty,tz);`
- Scalature (non uniformi)
  - `glScalef(sx,sy,sz);`
- Generica
  - `glMultMatrixf(float f*);`

Operazioni su Matrici

- Vista:
  - `void gluLookAt(eyex,eyey,eyez,`
    `cx,cy,cz,`
    `upx, upy, upz);`
- Proiezione:
  - `glOrtho2D(left, right, bottom, top);`
  - `void gluPerspective(fovy, aspect,`
    `zNear, zFar);`

Stack di Matrici

- Operazioni sullo stack
  - `glPushMatrix();`
  - `glPopMatrix();`
- Esempio di utilizzo:
  - glMatrixMode(GL_PROJECTION);
  - Push M_{prsp}
  - Carico una matrice di proiezione ortogonale per disegnare uno sfondo
  - Pop M_{prsp}
  - Disegno la scena sullo sfondo

Inviare triangoli

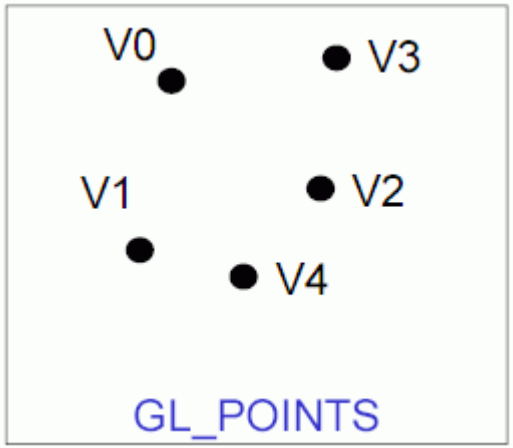
```
glBegin (GL_TRIANGLES);

glVertex3d(x1,y1,z1);
glVertex3d(x2,y2,z2);
glVertex3d(x3,y3,z3);

glVertex3d(x4,y4,z4);
glVertex3d(x5,y5,z5);
glVertex3d(x6,y6,z6);

glVertex3d(x7,y7,z7);
glVertex3d(x8,y8,z8);
glVertex3d(x9,y9,z9);
...
glEnd();
```

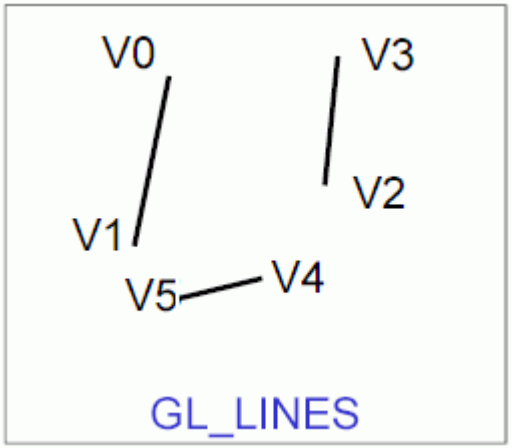
primo triangolo

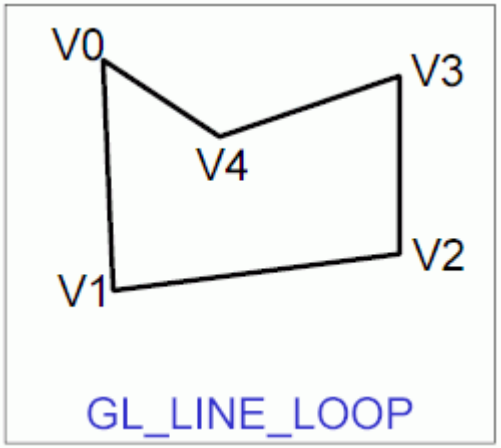
secondo triangolo

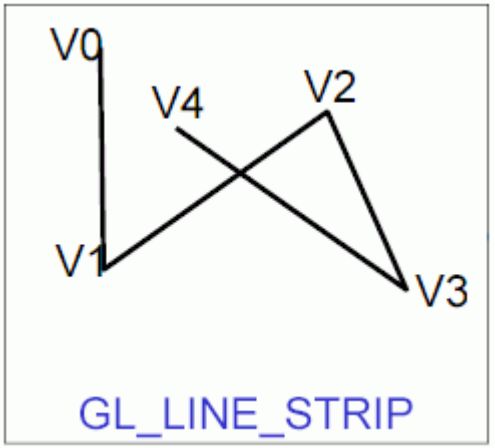
terzo triangolo

Inviare punti e linee

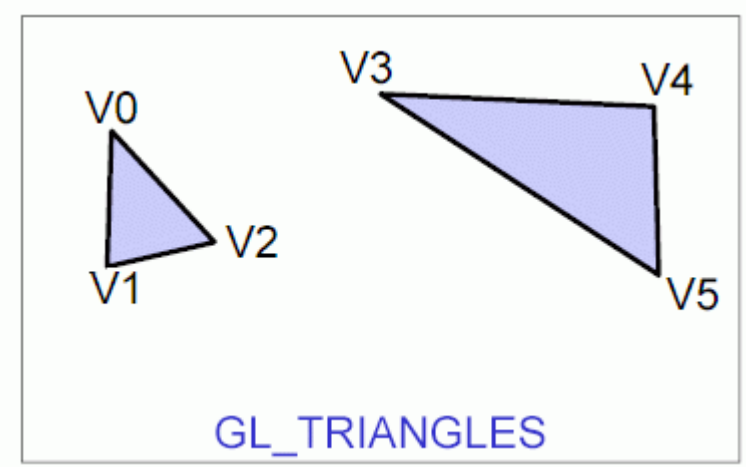


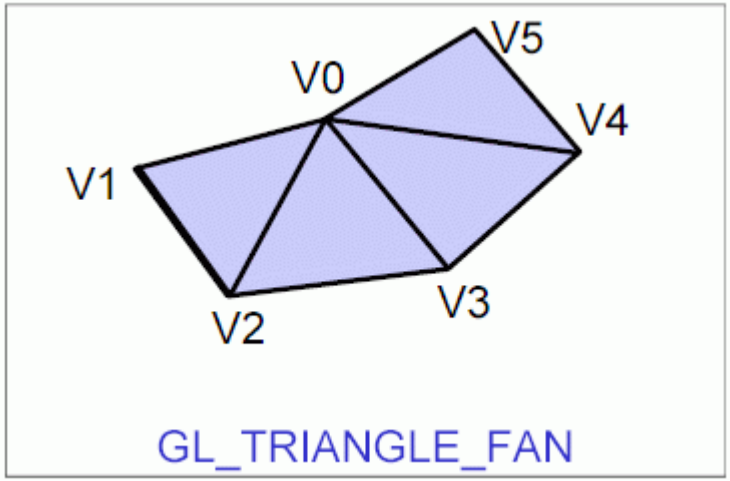


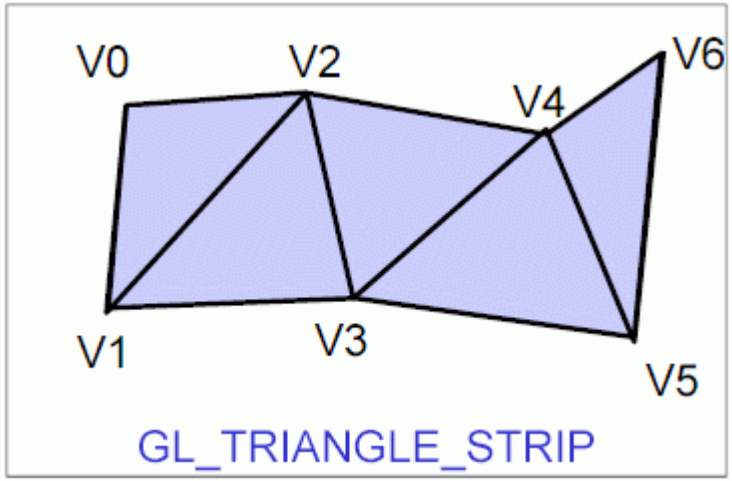




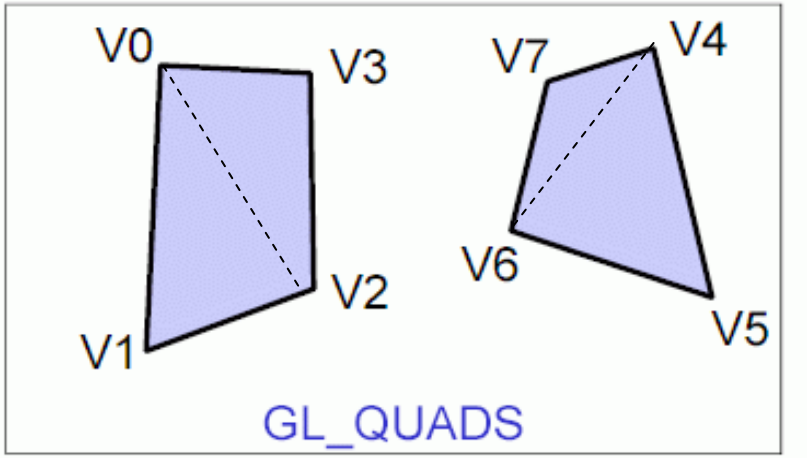
Inviare Fan e Strip

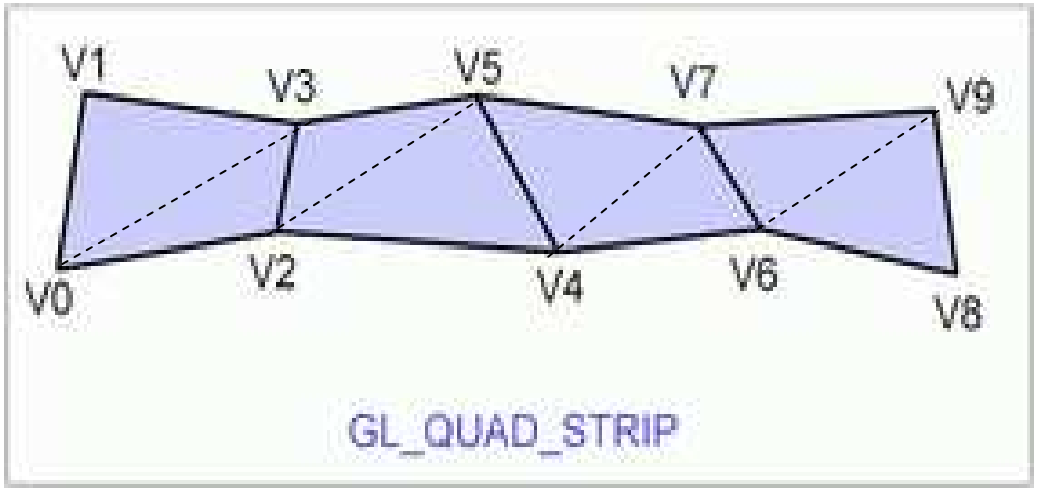






Inviare Quads





Approfondimenti

- Redbook
- Nehe’s Tutorial
- www.opengl.org

Domande?