Costruzione di Interfacce
Primi passi in OpenGL

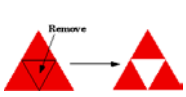
cignoni@iei.pi.cnr.it
http://vcg.iei.pi.cnr.it/~cignoni

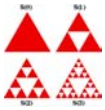
Introduzione

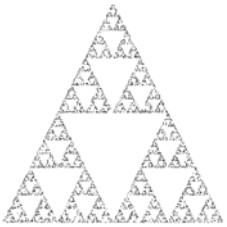
- Abbiamo visto
  - Cosa significa rendering
  - L'approccio object viewer
- Mettiamo in pratica qualcosa
- Scriviamo la prima applicazione che usa opengl
- Siccome non sappiamo ancora nulla ci terremo sul semplice.

Sierpinski gasket

- Si parte da un triangolo equilatero
- Si rimuove quello centrale
- Si procede ricorsivamente per I tre triangoli rimasti.



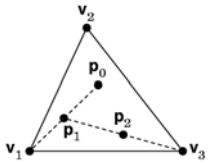


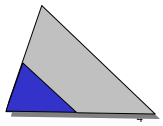


Sierpinski Gasket

- Approccio Generativo dell'insieme di punti P che appartengono al gasket:

p =punto a caso del triangolo
while true
  vi = vertice a caso del triangolo
  p = (p+vi)/2
  P=P U {p}





Stuttura del programma

Struttura classica dei programmi a linea di comando:

```
main()
{
    init();
    do_my_beautiful_algorithm();
    exit();
}
```

Non ha molto senso per i programmi con un'interfaccia utente.
Come avviene il processo di interazione tra l'utente e l'applicazione?

Event driven programming

- Gestione interazione applicazione-utente tramite *callback (message handlers ecc)*
  - funzioni che sono attivate in risposta a vari eventi (messaggi) gestiti dal sistema operativo (pressione di un tasto del mouse o della tastiera, reshape della finestra, necessita' di ridisegnare il contenuto della finestra ecc)
  - Il flusso principale dell'applicazione e' in mano all'utente o meglio al sistema operativo.

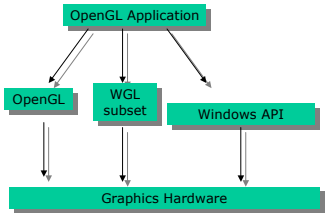
Approccio minimale per fare grafica

Per quello che ci riguarda la cosa piu' importante e' la gestione dell'evento:
"*Necessita' di disegnare il contenuto della finestra*"

Che fondamentalmente utilizzera' OpenGL.

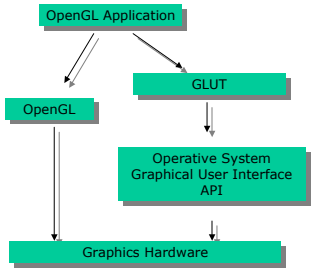
7

Struttura Windows OpenGL App



8

Struttura Applicazione OpenGL+Glut



9

GLUT

- GLUT is a window system independent toolkit for writing OpenGL programs.
- It implements a simple *portable* windowing application programming interface (API) for OpenGL.
- GLUT provides a portable API so you can write a single OpenGL program that works on both Win32 PCs and X11 workstation
- http://www.opengl.org/Documentation/GLUT.html

10

Schema Minimo Applicazione Glut

- Inizializzare
  - glutInit(&argc, argv)
- Definire e aprire una finestra
  - glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA | GLUT_DEPTH);
  - glutInitWindowSize(sizex,sizey);
  - glutCreateWindow(char *title)
- preparare le funzioni callback che SO invocherà
  - glutDisplayFunc(myRedrawFunc)
- Passare al SO/toolkit il controllo.
  - glutMainLoop()

11

La minima applicazione glut

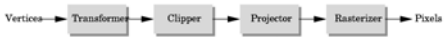
```
#include<stdio.h>
#include<GL/glut.h>

void myRedrawFunc()
{
   glClear(GL_COLOR_BUFFER_BIT);
        // Draw something
}
int main(int argc, char *argv[])
{
   glutInit(&argc, argv);
   glutCreateWindow("sierp");
   glutDisplayFunc(myRedrawFunc);
   glutMainLoop();   //Passare al SO il controllo.
}
```

12

Dove si disegna?

- Ricordate la pipeline di rendering



- La prima cosa che fa il renderer e' di spostare tutto nel sistema di riferimento della camera
- Poi taglia quel che non si vede
- Infine appiattisce il mondo sul piano di vista

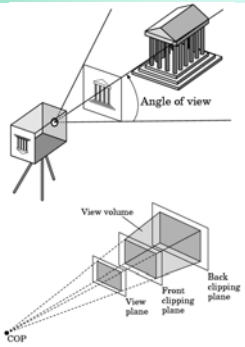
Trasformazioni e Clipping

- Noi non abbiamo definito nessuna trasformazione quindi vedremo solo quello che si trova *davanti* alla camera.
- Più precisamente vedremo quello che si trova nel *Volume di Vista:*

Porzione di spazio, nel sistema di riferimento della camera, che e' visibile dalla camera.

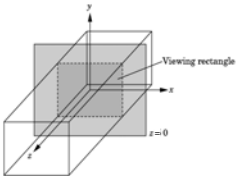
Volume di Vista

- Normalmente ci si aspetta che il volume di vista sia una piramide infinita.
- Per ragioni di praticità si aggiungono due piani (front and back o near and far) che ulteriormente delimitano lo spazio d'interesse, e quindi il volume di vista è un tronco di piramide.



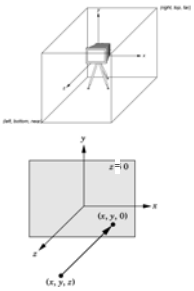
Proiezione Ortografica

- Se ci immaginiamo la camera posta ad una distanza infinita il volume di vista diventa un parallelepipedo.
- Questo genere di vista è detto ***ortogonale***

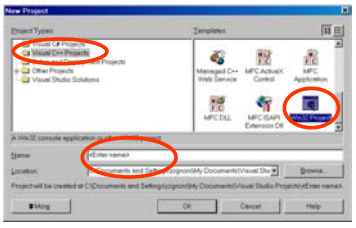


Proiezione Ortografica

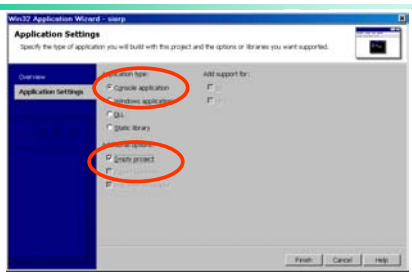
- In una proiezione ortografica tutti i punti nel volume di vista vengono semplicemente proiettati perpendicolamente sul piano di vista.



Creare il progetto

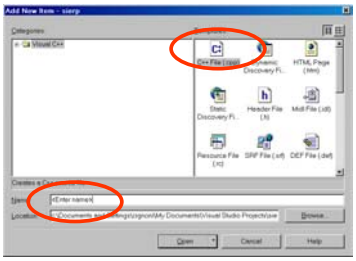


Setting progetto



19

Aggiungere un file al progetto



20

La minima applicazione glut

```
#include<stdio.h>
#include<GL/glut.h>

void myRedrawFunc()
{
   glClear(GL_COLOR_BUFFER_BIT);
        // Draw something
}
int main(int argc, char *argv[])
{
   glutInit(&argc, argv);
   glutCreateWindow("sierp");
   glutDisplayFunc(myRedrawFunc);
   glutMainLoop();   //Passare al SO il controllo.
}
```

21

Disegno del sierpinski set

- Questioni principali
  - Come si disegna un insieme di punti
  - Dove si disegna?

22

Come si disegna in OpenGL

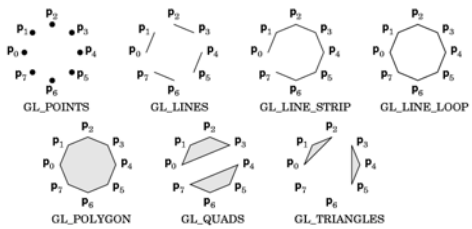
- Disegnare significa definire una scena da far passare nella pipeline

glBegin(Primitiva)
  Dati della primitiva (Coordinate vertici, e attributi vari)
glEnd()

- Le coordinate dei vertici si specificano con il comando
- glVertex*();

23

Primitive OpenGL



24

Note Su OpenGL

Note

- OpenGL e' il layer di base
- GLU e'un insieme di funzioni di utility costruite sopra OpenGL, piu'comode da usare
- GLUT e' il toolkit di interfaccia con il sistema operativo
- Wgl e GLx sono i sottoinsiemi di OpenGL che dipendono dal SO e che permettono di dire al SO ad esempio che l'interno di una certa finestra deve essere *adatto* a OpenGL. Per ora nascosto da GLUT
- Tutto quanto sopra e' C (e non C++).

25

GL syntax

- Tutte le funzioni di Opengl si chiamano
- glSomethingXXX
- Dove XXX specifica (numero) il tipo dei parametri:
- glColor3f(float, float, float)
  - f: float
  - d: double ecc.
- Non e' C++…

26

Disegnare il sierpinski gasket

- La generazione e' facile, quindi si può evitare di memorizzare e disegnare durante il processo di generazione.

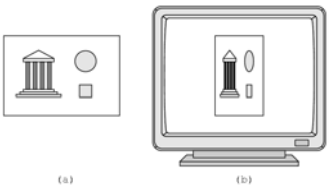
```
GLfloat triangle[3][2]={
      { -1.0f, -1.0f},{ 1.0f, -1.0f},{ 0.0f, 1.0f}       };

GLfloat p[2]={0.0f, 0.0f};
int i, j;
glClear(GL_COLOR_BUFFER_BIT);
glBegin(GL_POINTS);
for(j=0;j<20000;j++)
{
      i=rand()%3;
      p[0]=(p[0]+triangle[i][0])/2.0f;
      p[1]=(p[1]+triangle[i][1])/2.0f;
      glVertex2f(p[0],p[1]);
}
glEnd();
```

27

Gestione Reshape

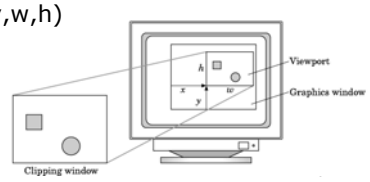
- Il comportamento di default e' che tutto il volume di vista viene mappato nella finestra.
- Aspect Ratio sbagliata



Viewport

- Di default si disegna su tutto la finestra, ma si puo specificare una sottporzione rettangolare della finestra (contesto) su cui si disegna

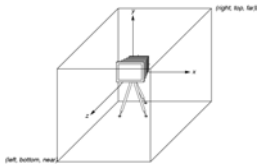
glViewport(x,y,w,h)



Adattare la camera alla finestra

- In opengl una vista ortogonale, si specifica definendo il view volume

glOrtho(left,right,bottom,top,near,far);



30

Adattare la camera alla finestra

- Il View Volume deve avere le stesse proporzioni della finestra
- Si usa un'altra callback quella in risposta all'evento di Reshape (aka resize) della finestra.

- `void myReshapeFunc(GLsizei w, GLsizei h)`
- `glutReshapeFunc(myReshapeFunc);`

Adattare la camera alla finestra

```
void myReshapeFunc(GLsizei w, GLsizei h)
{
  glMatrixMode (GL_PROJECTION);
  glLoadIdentity ();
  float ratio=(float)h/(float)w;
  glOrtho(-1,1,-ratio,ratio,-1,1);
  glViewport (0, 0, (GLsizei) w, (GLsizei) h);
  glMatrixMode (GL_MODELVIEW);
}
```