Corso di grafica 3 D con C++ e OpenGL

Alessandro Bugatti – ITIS B. Castelli

Sviluppo software in ambito grafico

- Contesto di programmazione molto specializzato e complesso
- Ricco di “sfide” informatiche (gestione memoria-prestazioni, periferiche, rete…)
- Richiede la conoscenza della fisica e della geometria
- Elemento trainante dell’evoluzione tecnologica del PC (esempio)
- Settore in continua espansione (mercati tipici film d'animazione, videogiochi, CAD,...)

Perché il C++

- Linguaggio maturo e stabile
- Adatto alla programmazione sia a basso livello che ad alto livello
- Supporto al paradigma di programmazione orientato agli oggetti
- Supporto alla programmazione generica
- Larga disponibilità di compilatori e librerie anche gratuiti

Perché OpenGL

- Libreria per la grafica 3D
- Libreria open, guidata da un comitato (ARB) formato da un insieme di aziende, con specifiche pubbliche
- Matura (esiste dal 1992, adesso è alla versione 4.4)
- Multipiattaforma (Windows, Linux/UNIX, MacOS, Android,…)

Applicazioni che utilizzano OpenGL

- Autodesk 3ds Max
- Blender
- Cinema 4D
- LightWave 3D
- Maya (“Il signore degli Anelli”, “Harry Potter”, …)
- Rhino3D, SketchUp, SolidThinking …

Cos'è OpenGL

- Nasce come libreria per la grafica 2D/3D per macchine high-end all'interno dei laboratori di Silicon Graphics
- È un'API grafica che espone al programmatore una serie di funzioni per la manipolazione di scene 3D
- Possiede delle estensioni per altri aspetti (gestione delle periferiche, finestre,…)

Cosa serve per creare un'applicazione

- Un compilatore C/C++
- La libreria OpenGL con gli header
- Una o più librerie per la gestione di ciò che non è pura grafica (window management, input handling, ecc.)
- Opzionalmente un ambiente di sviluppo
- Tools per la creazione delle "grafica" se si vuole fare qualcosa di professionale (che non verranno spiegati in questo corso)

Cosa utilizzeremo in Windows

- Ambiente di sviluppo Code::Blocks 10.05 che contiene:
  - compilatore g++ (TDM-2 mingw32) 4.4.1, porting del noto gcc in ambiente windows
  - Libreria OpenGL (e estensioni)
  - Libreria SDL per maneggiare tutti gli altri aspetti (inclusa nel pacchettone da me preparato,da usare per non avere problemi)

Differenze grafica 2D-3D

- La grafica 2D assomiglia ai cartoni animati di una volta (Biancaneve)
- L'effetto di movimento si ottiene spostando velocemente oggetti precedentemente disegnati
- Esiste il solo punto di vista frontale

Differenze grafica 2D-3D

- La grafica 3D assomiglia ai cartoni animati moderni realizzati al computer (Toy Story)

Aspetti tipici della grafica 3D

- Vengono creati dei modelli tridimensionali
- Esistono infiniti punti di vista, basta spostare la “telecamera”
- L’ambiente e tutto ciò che contiene devono comunque essere proiettati su una superficie bidimensionale
- Effetti come sfumature, ombre, luci, trasparenze ecc. vengono realizzati a costo zero (per il programmatore)

Pipeline grafica

- La geometria e le texture passano attraverso queste fasi
  - Determinazione della visibilità
    - Clippling
    - Culling
    - Occlusion testing
  - Determinazione della risoluzione (LOD)
  - Trasformazioni geometriche e illuminazione
  - Rasterizzazione

Come funziona OpenGL

- Per il programmatore OpenGL è un'API grafica che lo astrae dall'hardware sottostante, permettendo di dichiarare degli oggetti geometrici e le operazioni che devono essere effettuate su di essi.
- E' come una macchina a stati e ogni istruzione va a modificarne lo stato interno

OpenGL + SDL

- SDL (Simple DirectMedia Layer) è una libreria per l'accesso all'hardware che utilizzeremo insieme a OpenGL per gestire tutti gli aspetti non legati alla grafica
- Questo ci permetterà di creare programmi che gireranno su tutte le piattaforme che supportano SDL e OpenGL semplicemente ricompilandoli

Struttura di un programma

- Inizializzazione SDL
  - Inizializza la libreria
    - SDL_Init(SDL_INIT_VIDEO)
  - Abilita il double buffering
    - SDL_GL_SetAttribute(SDL_GL_DOUBLEBUFFER, 1);
  - Setta la modalità video
    - SDL_SetVideoMode(SCREEN_WIDTH, SCREEN_HEIGHT, SCREEN_BPP, videoFlags);

Struttura di un programma

- Inizializzazione OpenGL
- Vengono impostati una serie di parametri secondo ciò che si desidera ottenere
- Esempi:
  - Settare il colore di sfondo
  - Impostare i vari buffer
  - Abilitare alcuni test
  - Scegliere la prospettiva

Struttura di un programma

- Main loop: è un ciclo che termina generalmente quando si chiude il programma e al cui interno:
  - si processano gli eventi (input dell'utente, comunicazioni di rete, …)
  - si eseguono altri task (AI, simulazione del modello fisico,…)
  - si disegna a video

Disegno in immediate mode

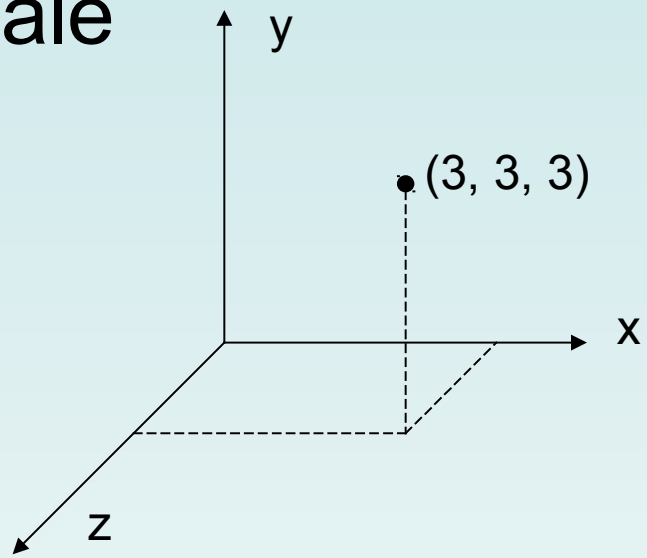
- Vengono date una serie di istruzioni fra un blocco di inizio e uno di fine, che vengono eseguite immediatamente

```
glBegin(MODE);
.....
.....
glEnd();
```

ISTRUZIONI

Disegno in immediate mode

- Le istruzioni possono rappresentare i vertici di una figura, espressi con le coordinate x,y,z in un piano cartesiano tridimensionale



Disegno in immediate mode

- A seconda del *mode* prescelto vengono interpretate le istruzioni in modo diverso
- Esistono 10 *mode*:

  GL_POINTS, **GL_LINES**, **GL_LINE_STRIP**, **GL_LINE_LOOP**, **GL_TRIANGLES**, **GL_TRIANGLE_STRIP**, **GL_TRIANGLE_FAN**, **GL_QUADS**, **GL_QUAD_STRIP**, **GL_POLYGON**.

OpenGL Mode

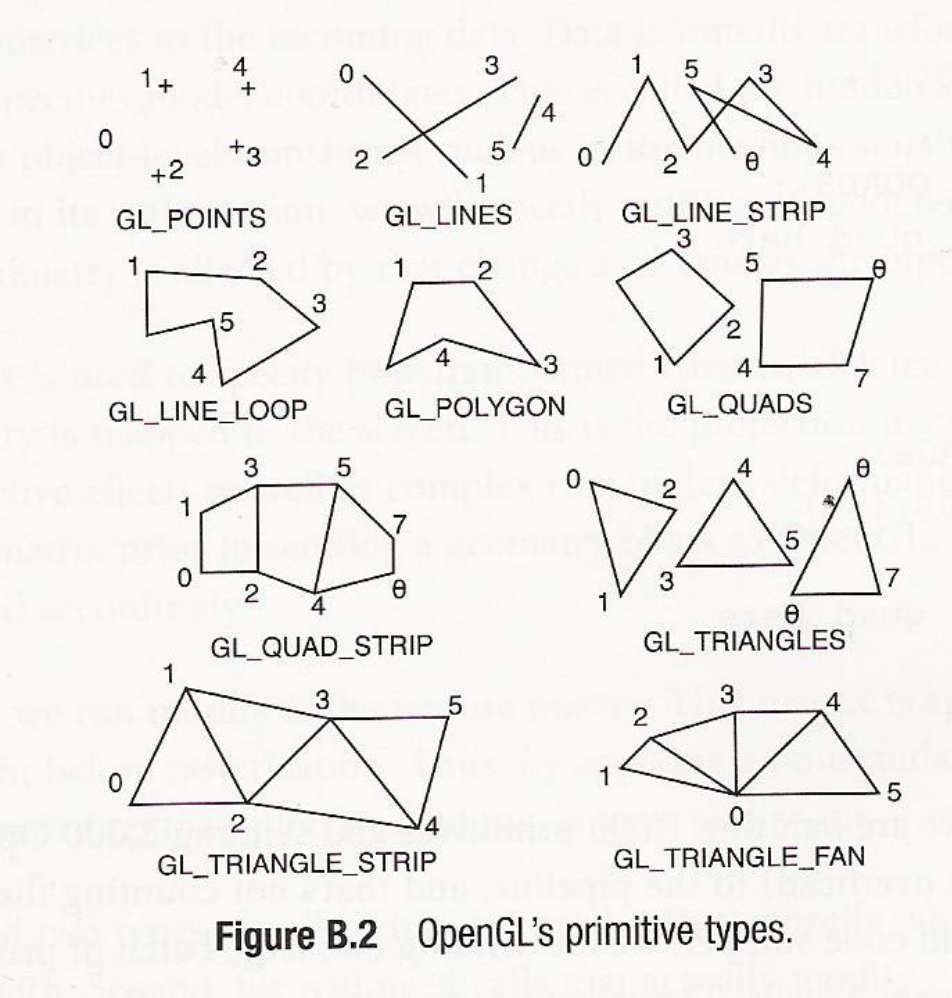


Figure B.2 OpenGL's primitive types.

Colorare una figura

- I colori possono essere rappresentati in formato RGB o RGBA
- L'istruzione glColor (in tutte le sue versioni) modifica il colore di tutte le primitive disegnate da quell'istruzione in poi
- A seconda dello shade model viene applicata un'interpolazione fra i colori

1990

La prima immagine che ho visto su un computer, fotografia con risoluzione 320x240

2014

PovRay, Hall of Fame (hof.povray.org), rendering fotorealistico