Grafica pe calculator – Preliminarii

Mihai-Sorin Stupariu

Sem. al II-lea, 2022 - 2023

De ce un curs de grafică? / Aplicații?

- Filme.

De ce un curs de grafică? / Aplicații?

- Filme.
- Dezvoltarea de jocuri.

De ce un curs de grafică? / Aplicații?

- Filme.
- Dezvoltarea de jocuri.
- Imagistică medicală.

Ce înseamnă un curs de grafică?

- **Abordări posibile:**

Ce înseamnă un curs de grafică?

- **Abordări posibile:**
  - background + dezvoltare "low level" (de exemplu *OpenGL*)

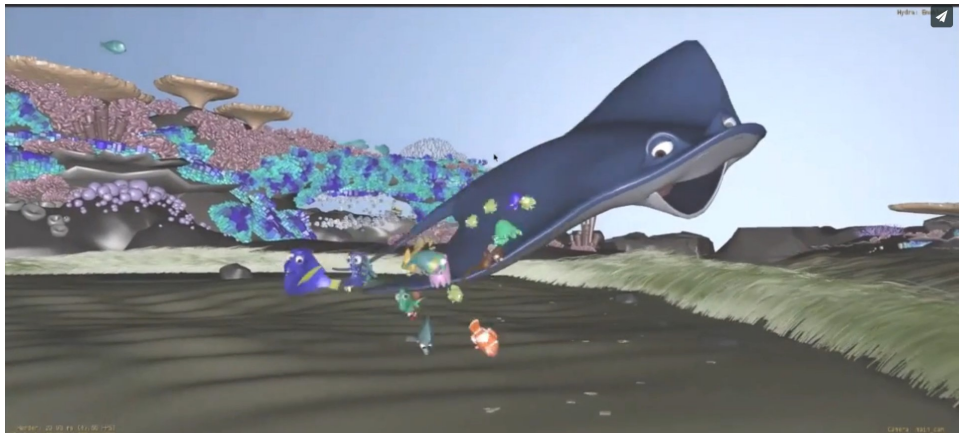


Figura: Sursa: https://www.iamag.co/real-time-graphics-in-pixar-film-production/

Ce înseamnă un curs de grafică?

- **Abordări posibile:**
  - background + dezvoltare “low level” (de exemplu *OpenGL*)

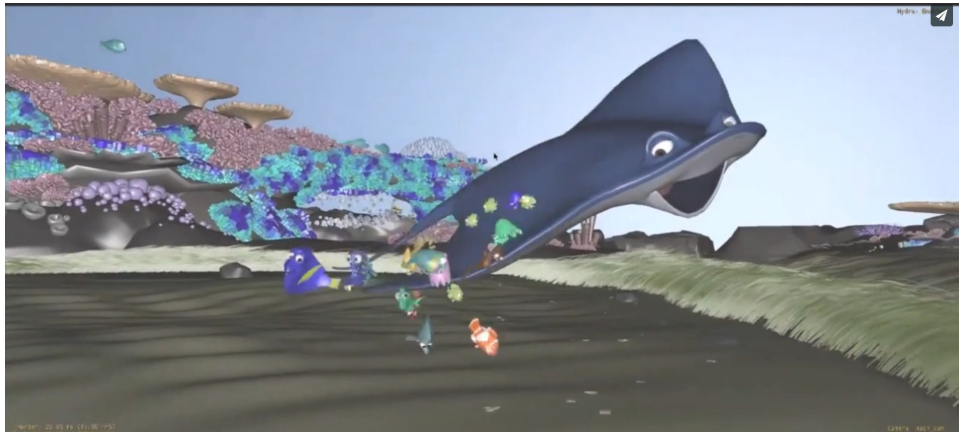
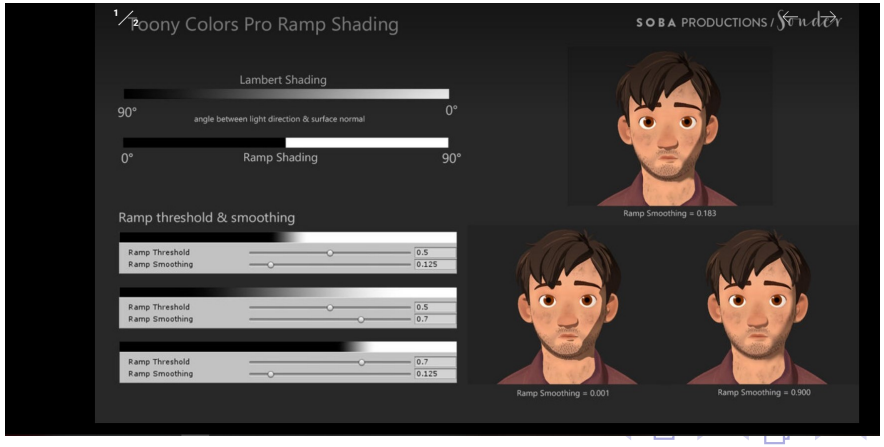


Figura: Sursa: https://www.iamag.co/real-time-graphics-in-pixar-film-production/

  - tehnologii dedicate (de exemplu *Unity*)



Cunoștințe necesare?

- cunoștințe elementare de programare în C++ (inclusiv utilizarea unui mediu de dezvoltare integrat) - vom folosi Microsoft Visual Studio;
- elemente de bază de Algebră liniară și Geometrie analitică.

Ce teme vom aborda la curs+laborator? Care ar fi obiectivul?

- Primitive grafice

Ce teme vom aborda la curs+laborator? Care ar fi obiectivul?

- Primitive grafice
- Transformări

Ce teme vom aborda la curs+laborator? Care ar fi obiectivul?

- Primitive grafice
- Transformări
- Texturare

Ce teme vom aborda la curs+laborator? Care ar fi obiectivul?

- Primitive grafice
- Transformări
- Texturare
- Reprezentarea scenelor 3D

Ce teme vom aborda la curs+laborator? Care ar fi obiectivul?

- Primitive grafice
- Transformări
- Texturare
- Reprezentarea scenelor 3D
- Iluminare

Ce teme vom aborda la curs+laborator? Care ar fi obiectivul?

- Primitive grafice
- Transformări
- Texturare
- Reprezentarea scenelor 3D
- Iluminare
- Efecte vizuale

Ce teme vom aborda la curs+laborator? Care ar fi obiectivul?

- Primitive grafice
- Transformări
- Texturare
- Reprezentarea scenelor 3D
- Iluminare
- Efecte vizuale
- Ideal: la finalul cursului să puteți dezvolta o aplicație complexă 3D

Elemente generale

- OpenGL este o interfață de programare (Application Programming Interface API)

Elemente generale

- OpenGL este o interfață de programare (Application Programming Interface API)
- Pentru a funcționa are nevoie de o serie de biblioteci; versiunea suportată depinde de placa grafică a calculatorului - se poate folosi codul sursa `00_test_version.cpp`

Elemente generale

- OpenGL este o interfață de programare (Application Programming Interface API)
- Pentru a funcționa are nevoie de o serie de biblioteci; versiunea suportată depinde de placa grafică a calculatorului - se poate folosi codul sursa `00_test_version.cpp`
- OpenGL **nu** este un limbaj de programare (există GLSL)

Elemente generale

- OpenGL este o interfață de programare (Application Programming Interface API)
- Pentru a funcționa are nevoie de o serie de biblioteci; versiunea suportată depinde de placa grafică a calculatorului - se poate folosi codul sursa `00_test_version.cpp`
- OpenGL **nu** este un limbaj de programare (există GLSL)
- Arhitectura de tip *client-server* (CPU-GPU) - element cheie: "comunicarea" dintre cele două componente *hardware*

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; “open standard”

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; “open standard”
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; "open standard"
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)
- 1997: versiunea 1.1

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; "open standard"
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)
- 1997: versiunea 1.1
- 2002/2003: versiunile 1.4, 1.5; GLSL (GL Shading Language) apare ca o extensie a funcționalității de bază

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; "open standard"
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)
- 1997: versiunea 1.1
- 2002/2003: versiunile 1.4, 1.5; GLSL (GL Shading Language) apare ca o extensie a funcționalității de bază
- 2004: **OpenGL 2.0; GLSL 1.10.59 inclus în "core"**

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; "open standard"
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)
- 1997: versiunea 1.1
- 2002/2003: versiunile 1.4, 1.5; GLSL (GL Shading Language) apare ca o extensie a funcționalității de bază
- 2004: **OpenGL 2.0; GLSL 1.10.59 inclus în "core"**
- 2008: OpenGL 3.0: conceptul de "funcționalități depreciate", legate de modul de redare imediat; mecanismul de depreciere a fost implementat / extins în 2009, când au fost lansate versiunile 3.1 și 3.2

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; "open standard"
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)
- 1997: versiunea 1.1
- 2002/2003: versiunile 1.4, 1.5; GLSL (GL Shading Language) apare ca o extensie a funcționalității de bază
- 2004: **OpenGL 2.0; GLSL 1.10.59 inclus în "core"**
- 2008: OpenGL 3.0: conceptul de "funcționalități depreciate", legate de modul de redare imediat; mecanismul de depreciere a fost implementat / extins în 2009, când au fost lansate versiunile 3.1 și 3.2
- 2010: OpenGL 3.3 (hardware care suportă Direct3D 10); OpenGL 4.0 (hardware care suportă Direct3D 11); numerotarea versiunilor de GLSL este "sincronizată" (GLSL v. 3.30.6, respectiv v. 4.00.9)

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; "open standard"
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)
- 1997: versiunea 1.1
- 2002/2003: versiunile 1.4, 1.5; GLSL (GL Shading Language) apare ca o extensie a funcționalității de bază
- 2004: **OpenGL 2.0; GLSL 1.10.59 inclus în "core"**
- 2008: OpenGL 3.0: conceptul de "funcționalități depreciate", legate de modul de redare imediat; mecanismul de depreciere a fost implementat / extins în 2009, când au fost lansate versiunile 3.1 și 3.2
- 2010: OpenGL 3.3 (hardware care suportă Direct3D 10); OpenGL 4.0 (hardware care suportă Direct3D 11); numerotarea versiunilor de GLSL este "sincronizată" (GLSL v. 3.30.6, respectiv v. 4.00.9)
- 2014 OpenGL 4.5 (respectiv GLSL 4.50)

Scurt istoric

- 1992: versiunea 1.0 a OpenGL (derivat din IRIS GL), lansat de SGI; "open standard"
- 1992: OpenGL Architecture Review Board (elaborarea specificațiilor)
- 1997: versiunea 1.1
- 2002/2003: versiunile 1.4, 1.5; GLSL (GL Shading Language) apare ca o extensie a funcționalității de bază
- 2004: **OpenGL 2.0; GLSL 1.10.59 inclus în "core"**
- 2008: OpenGL 3.0: conceptul de "funcționalități depreciate", legate de modul de redare imediat; mecanismul de depreciere a fost implementat / extins în 2009, când au fost lansate versiunile 3.1 și 3.2
- 2010: OpenGL 3.3 (hardware care suportă Direct3D 10); OpenGL 4.0 (hardware care suportă Direct3D 11); numerotarea versiunilor de GLSL este "sincronizată" (GLSL v. 3.30.6, respectiv v. 4.00.9)
- 2014 OpenGL 4.5 (respectiv GLSL 4.50)
- 2017 OpenGL 4.6 (respectiv GLSL 4.60)

Biblioteci utilizate de OpenGL și funcții asociate

- bibliotecă fundamentală (core library): este independentă de platforma pe care se lucrează; funcțiile corespunzătoare au prefixul `gl` (de exemplu: `glClearColor ( ); glFlush ( ); glVertex ( ); glColor ( );`
`glBegin ( )`, etc.
- GLU (OpenGL Utility): conține mai ales proceduri / funcții legate de proiecție, precum și funcții pentru conice și cuadrice; funcțiile asociate au prefixul `glu`
- GLUT (OpenGL Utility Toolkit) / freeglut: bibliotecă *dependentă de sistem*, utilizată pentru a realiza fereastra de vizualizare; poate interacționa cu sisteme de operare bazate pe ferestre de vizualizare; funcțiile specifice au prefixul `glut`

Resurse

- Site-ul oficial OpenGL

Resurse

- Site-ul oficial OpenGL
- Cărți: D. Hearn si M. Baker, Computer Graphics with OpenGL, Prentice Hall, 2003;
  D. Shreiner, M.Woo, J. Neider si T. Davis. OpenGL Programming Guide, Addison Wesley; coduri sursă.

Resurse

- Site-ul oficial OpenGL
- Cărți: D. Hearn si M. Baker, Computer Graphics with OpenGL, Prentice Hall, 2003;
  D. Shreiner, M.Woo, J. Neider si T. Davis. OpenGL Programming Guide, Addison Wesley; coduri sursă.
- Cărți / tutoriale online: D. Eck, Introduction to Computer Graphics; http://nehe.gamedev.net/

Exemplu de program care utilizează OpenGL

Codul sursă `01_C_1_exemplu.cpp`

```
// Directive preprocesare
```

Exemplu de program care utilizează OpenGL

Codul sursă `01_C_1_exemplu.cpp`

```
// Directive preprocesare
// Procedură inițializare – init
```

Exemplu de program care utilizează OpenGL

Codul sursă `01_C_1_exemplu.cpp`

```
// Directive preprocesare
// Procedură inițializare – init
// Procedură desen – desen
```

Exemplu de program care utilizează OpenGL

Codul sursă `01_C_1_exemplu.cpp`

```
// Directive preprocesare
// Procedură inițializare – init
// Procedură desen – desen
// Main
```

Exemplu de program care utilizează OpenGL

Codul sursă `01_C_1_exemplu.cpp`

```
// Directive preprocesare
// Procedură inițializare – init
// Procedură desen – desen
// Main
    // Inițializări GLUT
    // Generare fereastră
    // Apelare procedură inițializare
    // Apelare procedură desen
    // Apelare glutMainLoop
```

Exemplu de program care utilizează OpenGL

```
#include <windows.h> // biblioteci care urmeaza sa fie incluse.
#include <gl/freeglut.h> // nu trebuie uitat freeglut.h
void init (void) // initializare ecran de vizualizare
{
    glClearColor (1.0, 1.0, 1.0, 0.0); // culoare de fond ecran
    glMatrixMode (GL_PROJECTION); // reprez.2D; pr.  ortogonala
    gluOrtho2D (0.0, 400.0, 0.0, 300.0);
}
void desen (void) // procedura desenare
{
    glColor3f (0.0, 0.0, 1.0); // culoarea punctelor:  albastru
    glBegin (GL_POINTS);
       glVertex2i (20, 20);
       glVertex2i (21, 21);
       glVertex2i (22, 22);
       glVertex2i (23, 23);
       glVertex2i (24, 24);
       glVertex2i (100, 100);
    glEnd ( );
```

Exemplu (continuare)

```
    glColor3f (1.0, 0.0, 0.0); // culoarea liniei frante:  rosu
    glBegin (GL_LINE_STRIP);
       glVertex2i (0,0);
       glVertex2i (200, 100);
       glVertex2i (300, 120);
    glEnd ( );
    glFlush ( );
}
void main (int argc, char** argv)
{
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowPosition (100, 100); // pozitia initiala
    glutInitWindowSize (800, 600); // dimensiunile ferestrei
    glutCreateWindow ("Puncte & Segmente"); // titlul ferestrei
    init ( );
    glClear (GL_COLOR_BUFFER_BIT);
    glutDisplayFunc (desen);
    glutMainLoop ( );
}
```

Elemente importante

- Directive preprocesare

```
#include <windows.h> // biblioteci care urmeaza sa fie incluse.
#include <gl/freeglut.h> // nu trebuie uitat freeglut.h
```

Elemente importante

- Directive preprocesare

```
#include <windows.h> // biblioteci care urmeaza sa fie incluse.
#include <gl/freeglut.h> // nu trebuie uitat freeglut.h
```

- Procedură inițializare `init`

```
void init (void) // initializare ecran de vizualizare
{
glClearColor (1.0, 1.0, 1.0, 0.0); // culoarea de fond a ecranului
glMatrixMode (GL_PROJECTION); // reprezentare 2D; proiectie ortogonala
gluOrtho2D (0.0, 400.0, 0.0, 300.0);
}
```

Elemente importante

- Directive preprocesare

```
#include <windows.h> // biblioteci care urmeaza sa fie incluse.
#include <gl/freeglut.h> // nu trebuie uitat freeglut.h
```

- Procedură inițializare `init`

```
void init (void) // initializare ecran de vizualizare
{
glClearColor (1.0, 1.0, 1.0, 0.0); // culoarea de fond a ecranului
glMatrixMode (GL_PROJECTION); // reprezentare 2D; proiectie ortogonala
gluOrtho2D (0.0, 400.0, 0.0, 300.0);
}
```

- Procedură desen `desen`

Elemente importante

- Directive preprocesare

```
#include <windows.h> // biblioteci care urmeaza sa fie incluse.
#include <gl/freeglut.h> // nu trebuie uitat freeglut.h
```

- Procedură inițializare `init`

```
void init (void) // initializare ecran de vizualizare
{
glClearColor (1.0, 1.0, 1.0, 0.0); // culoarea de fond a ecranului
glMatrixMode (GL_PROJECTION); // reprezentare 2D; proiectie ortogonala
gluOrtho2D (0.0, 400.0, 0.0, 300.0);
}
```

- Procedură desen `desen`
- Main

Elemente importante

- Directive preprocesare

```
#include <windows.h> // biblioteci care urmeaza sa fie incluse.
#include <gl/freeglut.h> // nu trebuie uitat freeglut.h
```

- Procedură inițializare `init`

```
void init (void) // initializare ecran de vizualizare
{
glClearColor (1.0, 1.0, 1.0, 0.0); // culoarea de fond a ecranului
glMatrixMode (GL_PROJECTION); // reprezentare 2D; proiectie ortogonala
gluOrtho2D (0.0, 400.0, 0.0, 300.0);
}
```

- Procedură desen `desen`
- Main
  - Inițializări GLUT
  - Generare fereastră
  - Apelare procedură inițializare
  - Apelare procedură desen
  - Apelare MainLoop