

Primitive grafice

Mihai-Sorin Stupariu

Sem. I, 2022 - 2023

Spațiul culorilor

Vârfuri

Primitive grafice - generalități

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) — **Cubul RGB**.

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) — **Cubul RGB**.
- ▶ În OpenGL o culoare este indicată prin:

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) — **Cubul RGB**.
- ▶ În OpenGL o culoare este indicată prin:
 - în OpenGL “vechi” folosind funcția

```
glColor* ( );
```

— “deprecated”

Sufixul * poate indica:

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) – **Cubul RGB**.
- ▶ În OpenGL o culoare este indicată prin:
 - în OpenGL “vechi” folosind funcția

```
glColor* ( );
```

– “deprecated”

Sufixul * poate indica:

- dimensiunea n a spațiului de culori în care lucrăm, $n = 3$ (RGB) sau $n = 4$ (RGBA); A=factorul “alpha”, legat de opacitate/transparență.

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) – **Cubul RGB**.
- ▶ În OpenGL o culoare este indicată prin:
 - în OpenGL “vechi” folosind funcția

```
glColor* ( );
```

– “deprecated”

Sufixul * poate indica:

- dimensiunea n a spațiului de culori în care lucrăm, $n = 3$ (RGB) sau $n = 4$ (RGBA); A=factorul “alpha”, legat de opacitate/transparență.
- tipul de date utilizat, care poate fi:
 - i (integer) ($i \in \{0, 1, \dots, 255\}$)
 - f (float)
 - d (double) ($f, d \in [0.0, 1.0]$)

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) – **Cubul RGB**.
- ▶ În OpenGL o culoare este indicată prin:
 - în OpenGL “vechi” folosind funcția

```
glColor* ( );
```

– “deprecated”

Sufixul * poate indica:

- dimensiunea n a spațiului de culori în care lucrăm, $n = 3$ (RGB) sau $n = 4$ (RGBA); A=factorul “alpha”, legat de opacitate/transparență.
- tipul de date utilizat, care poate fi:
 - i (integer) ($i \in \{0, 1, \dots, 255\}$)
 - f (float)
 - d (double) ($f, d \in [0.0, 1.0]$)
- (opțional) posibila formă vectorială, indicată prin sufixul v.

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) – **Cubul RGB**.
- ▶ În OpenGL o culoare este indicată prin:

- în OpenGL “vechi” folosind funcția

```
glColor* ( );
```

– “deprecated”

Sufixul * poate indica:

- dimensiunea n a spațiului de culori în care lucrăm, $n = 3$ (RGB) sau $n = 4$ (RGBA); A=factorul “alpha”, legat de opacitate/transparență.
- tipul de date utilizat, care poate fi:
 - i (integer) ($i \in \{0, 1, \dots, 255\}$)
 - f (float)
 - d (double) ($f, d \in [0.0, 1.0]$)
- (opțional) posibila formă vectorială, indicată prin sufixul v.
- în OpenGL “nou”: culorile sunt manevrate într-un mod asemănător vârfulor (folosind VBO), odată cu acestea (attribute ale vârfulor).

Spațiul culorilor

- ▶ Culorile sunt obținute combinând intensitățile de pe trei canale: R (red); G (green); B (blue) – **Cubul RGB**.

- ▶ În OpenGL o culoare este indicată prin:

- în OpenGL “vechi” folosind funcția

```
glColor* ( );
```

– “deprecated”

Sufixul * poate indica:

- dimensiunea n a spațiului de culori în care lucrăm, $n = 3$ (RGB) sau $n = 4$ (RGBA); A=factorul “alpha”, legat de opacitate/transparență.
- tipul de date utilizat, care poate fi:
 - i (integer) ($i \in \{0, 1, \dots, 255\}$)
 - f (float)
 - d (double) ($f, d \in [0.0, 1.0]$)
- (opțional) posibila formă vectorială, indicată prin sufixul v.
- în OpenGL “nou”: culorile sunt manevrate într-un mod asemănător vârfurilor (folosind VBO), odată cu acestea (atribute ale vârfurilor).
- elementul comun este faptul că, în ambele cazuri, culorile conțin informații legate de canalele R, G, B, precum și de canalul A (factorul α , legat de opacitate).

Vârfuri - funcții pentru indicare

Primitivele grafice sunt trasate cu ajutorul **vârfurilor** (*entități abstracte, a nu fi confundate cu punctele!*). În OpenGL un vârf este definit:

Vârfuri - funcții pentru indicare

Primitivele grafice sunt trasate cu ajutorul **vârfurilor** (*entități abstracte, a nu fi confundate cu punctele!*). În OpenGL un vârf este definit:

- în OpenGL “vechi” cu ajutorul funcției

```
glVertex* ( );
```

— “deprecated”

Vârfuri - funcții pentru indicare

Primitivele grafice sunt trasate cu ajutorul **vârfurilor** (*entități abstracte, a nu fi confundate cu punctele!*). În OpenGL un vârf este definit:

- în OpenGL “vechi” cu ajutorul funcției

```
glVertex* ( );
```

— “deprecated”

- în OpenGL “nou”:
vârfurile sunt stocate în matrice;
împachetate în VAO (Vertex Array Objects);
trimise plăcii grafice sub formă de VBO (Vertex Buffer Objects).

Caracteristici ale vârfurilor

- Unui vârf îi sunt asociate:

Caracteristici ale vârfurilor

- ▶ Unui vârf îi sunt asociate:
 - ▶ coordonate (fac parte din definiție),

Caracteristici ale vârfurilor

- ▶ Unui vârf îi sunt asociate:
 - ▶ coordonate (fac parte din definiție),
 - ▶ o culoare (v. secțiunea Culori...),

Caracteristici ale vârfurilor

- ▶ Unui vârf îi sunt asociate:
 - ▶ coordonate (fac parte din definiție),
 - ▶ o culoare (v. secțiunea Culori...),
 - ▶ o normală (legată de funcții de iluminare),

Caracteristici ale vârfurilor

- ▶ Unui vârf îi sunt asociate:
 - ▶ coordonate (fac parte din definiție),
 - ▶ o culoare (v. secțiunea Culori...),
 - ▶ o normală (legată de funcții de iluminare),
 - ▶ coordonate de texturare

Caracteristici ale vârfurilor

- ▶ Unui vârf îi sunt asociate:
 - ▶ coordonate (fac parte din definiție),
 - ▶ o culoare (v. secțiunea Culori...),
 - ▶ o normală (legată de funcții de iluminare),
 - ▶ coordonate de texturare
- ▶ În OpenGL “vechi”: pentru o anumită caracteristică, este considerată valoarea *curentă* a respectivei caracteristici. Altfel spus, ea trebuie indicată în codul sursă înaintea vârfului.

Funcții pentru primitive

Vârfurile sunt utilizate pentru trasarea primitivelor grafice. Funcțiile folosite sunt diferite pentru cele două moduri de randare:

- în OpenGL “vechi”, o funcție de tipul `glVertex ()` poate fi apelată într-un cadru de tip

```
glBegin (*);
```

```
glEnd;
```

— “deprecated”

(unde * reprezintă tipul de primitivă generat);

Funcții pentru primitive

Vârfurile sunt utilizate pentru trasarea primitivelor grafice. Funcțiile folosite sunt diferite pentru cele două moduri de randare:

- în OpenGL “vechi”, o funcție de tipul `glVertex ()` poate fi apelată într-un cadru de tip

```
glBegin (*);
```

```
glEnd;
```

— “deprecated”

(unde * reprezintă tipul de primitivă generat);

- în OpenGL “nou” este utilizată o funcție de tipul

```
glDrawArrays (GLenum mode, GLint first, GLint count);
```

unde

mode: tipul primitivei;

first: primul vârf;

count: câte vârfuri se iau în considerare.

Tipuri de primitive

► Puncte: `GL_POINTS`

Tipuri de primitive

- ▶ Puncte: `GL_POINTS`
- ▶ Segmente de dreaptă: `GL_LINES`, `GL_LINE_STRIP`, `GL_LINE_LOOP`

Tipuri de primitive

- ▶ Puncte: `GL_POINTS`
- ▶ Segmente de dreaptă: `GL_LINES`, `GL_LINE_STRIP`, `GL_LINE_LOOP`
- ▶ Triunghiuri: `GL_TRIANGLES`, `GL_TRIANGLE_STRIP`,
`GL_TRIANGLE_FAN`

Tipuri de primitive

- ▶ Puncte: `GL_POINTS`
- ▶ Segmente de dreaptă: `GL_LINES`, `GL_LINE_STRIP`, `GL_LINE_LOOP`
- ▶ Triunghiuri: `GL_TRIANGLES`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`
- ▶ Dreptunghiuri: `GL_QUADS`, `GL_QUAD_STRIP`

Tipuri de primitive

- ▶ Puncte: `GL_POINTS`
- ▶ Segmente de dreaptă: `GL_LINES`, `GL_LINE_STRIP`, `GL_LINE_LOOP`
- ▶ Triunghiuri: `GL_TRIANGLES`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`
- ▶ Dreptunghiuri: `GL_QUADS`, `GL_QUAD_STRIP`
- ▶ Poligoane (convexe!): `GL_POLYGON`