

Texturare

Mihai-Sorin Stupariu

Sem. I, 2024 - 2025

Principii generale

Etape și funcții asociate

Coordonate de modelare și coordonate de texturare

Concluzii

Ce este o textură?

- ▶ În general este o entitate (imagine externă) / generată procedural “aplicată” pe primitivele randate.
- ▶ O textură are o structură discretă, ca o imagine = **structură de texeli**.
- ▶ Pentru a putea “lega” textura de imagine, aceasta are nevoie de **coordonate de texturare**. Prin referire la textura dată, acestea sunt între 0 și 1.

Elemente necesare

Folosirea unei biblioteci dedicate (de exemplu *SOIL – Simple OpenGL Image Library*) permite incarcarea rapida a unor texturi din fisiere avand formate standard, precum JPEG, PNG, etc.

Template-ul pus la dispoziție are integrate această bibliotecă (fișierele `lib` aferente și fișierul `SOIL.h` utilizat ca fișier de tip header in proiect.).

Etape - sinteză

Din punctul de vedere al implementării, trebuie urmărite o serie de etape:

Etape - sinteză

Din punctul de vedere al implementării, trebuie urmărite o serie de etape:

- ▶ indicarea, în VBO a legăturii dintre coordonatele vârfurilor (x, y, z, w) și coordonatele de texturare (s, t) pentru fiecare vârf (slide 5),

Etape - sinteză

Din punctul de vedere al implementării, trebuie urmărite o serie de etape:

- ▶ indicarea, în VBO a legăturii dintre coordonatele vârfurilor (x, y, z, w) și coordonatele de texturare (s, t) pentru fiecare vârf (slide 5),
- ▶ crearea unei texturi sau a unui obiect textură și specificarea caracteristicilor texturii, precum și indicarea modului în care textura este aplicată pe pixelii imaginii (slide 6),

Etape - sinteză

Din punctul de vedere al implementării, trebuie urmărite o serie de etape:

- ▶ indicarea, în VBO a legăturii dintre coordonatele vârfurilor (x, y, z, w) și coordonatele de texturare (s, t) pentru fiecare vârf (slide 5),
- ▶ crearea unei texturi sau a unui obiect textură și specificarea caracteristicilor texturii, precum și indicarea modului în care textura este aplicată pe pixelii imaginii (slide 6),
- ▶ activarea texturării / transmiterea către shader a obiectului textură ca variabilă uniformă (slide 18).

Asocierea coordonatelor de texturare pentru fiecare vârf

În funcția de creare a *VAO* / *VBO* sunt indicate, pentru fiecare vârf, coordonatele de texturare aferente (trebuie să existe o coerență între coordonatele vârfurilor și modul de alegere a coordonatelor de texturare). Aceste coordonate au asociată o locație specifică, urmând să devină *vertex attributes* (în shader-ul de vârfuri). În codul sursă `04_04_texturare.cpp` coordonatele de texturare au locația 2.

Template - funcția LoadTexture

Funcția `LoadTexture()` conține elementele necesare creării obiectului textură (slide 7 și următoarele), incluzând generarea și legarea texturii, precum și precizarea proprietăților acesteia.

Indicarea modului în care textura este aplicată pe pixelii imaginii (`glTexParameteri`, slide 13 și următoarele).

Nu trebuie uitată eliberarea memoriei și realocarea.

`LoadTexture()` poate fi apelată la inițializare.

Crearea unui obiect textură și specificarea proprietăților acestuia

Obiectele textură memorează date referitoare la textură, făcându-le accesibile imediat. Fiecărui obiect îi este asociată o singură textură. Sunt parcurși câțiva pași intermediari, descriși în continuare, împreună cu funcțiile OpenGL asociate.

Generarea numelor obiectelor textură

:

```
glGenTextures (n, *texNames);
```

Prin această funcție sunt *rezervați* n identificatori de textură în vectorul `texNames`, declarat explicit în procedura de inițializare `init`. Rezervarea are ca semnificație faptul că numele din `texNames` sunt marcate ca fiind utilizate, ele primesc caracteristici efective atunci când sunt prima dată legate cu funcțiile specifice. **Observație.** De menționat că funcția OpenGL cu efect contrar celei descrise este funcția de ștergere

```
glDeleteTextures (n, *texNames);
```

Crearea și utilizarea obiectelor textură

```
glBindTexture (target, id);
```

Aici `target` este parametrul care descrie dimensiunea texturii; dacă este vorba de o textură 2-dimensională acesta este `GL_TEXTURE_2D`, iar `id` este numele texturii. Prin această funcție este *creat* un nou obiect textură, cu numele `id`. La apelarea lui `glBindTexture (...)`; în cadrul funcției de desenare, textura este legată de obiectul desenat, ceea ce este corelat cu corespondența dintre coordonatele de texturare și cele de modelare. De fapt, `glBindTexture (...)`; precizează obiectul textură activ.

Specificarea caracteristicilor unei texturi 1D

Definirea unei texturi 1D se face apelând funcția

```
glTexImage1D (.....);
```

Funcția `glTexImage1D` are parametrii

- ▶ `target`: tipul de textură (`GL_TEXTURE_1D`);
- ▶ `level`: nivelul de texturare (0);
- ▶ `intformat`: numărul valorilor (de culoare) utilizate pentru fiecare pixel; în cazul codului RGBA acest număr este 4;
- ▶ `width`: lățimea este o putere a lui 2; trebuie să fie coerentă și consistentă cu modul în care a fost definită textura;
- ▶ `border`: este un număr egal cu 0,1 sau 2 și indică numărul pixelilor de pe frontieră;
- ▶ `format`: poate fi o constantă simbolică de forma `GL_RGB`; `GL_RGBA`;
- ▶ `type`: tipul este indicat printr-o constantă simbolică `GL_UNSIGNED_BYTE`; `GL_BYTE`;
- ▶ `texels`: se indică unde este localizată textura.

Specificarea caracteristicilor unei texturi 2D

```
glTexImage2D (.....);
```

Funcția `glTexImage2D` are parametrii

- ▶ `target`: tipul de textură (`GL_TEXTURE_2D`);
- ▶ `level, intformat`: similar cu cazul 1D;
- ▶ `width, height`: lățimea și înălțimea texturii;
- ▶ `border`: dimensiunea frontierei (b_w, b_h);
- ▶ `format, type, texels`: similar cu cazul 1D.

Subtexturi

Trebuie menționat faptul că pot fi definite și subtexturi, prin funcția

```
glTexSubimage2D (.....);
```

având parametrii similari

- ▶ target: tipul de textură (GL_TEXTURE_2D);
- ▶ level: similar cu funcția glTexImage2D;
- ▶ xoffset, yoffset: sunt numere întregi, pozitive, care precizează unde anume este așezată subtextura în structura de texeli (cu convenția că (0,0) este în stânga jos);
- ▶ width, height: lățimea și înălțimea subtexturii;
- ▶ format, type, texels: similar cu funcția glTexImage2D.

Semnificația acestor funcții este că se definește o structură prin care este înlocuită parțial / total o porțiune a texturii active (curente).

Precizarea unor reguli referitoare la modul în care texeli sunt așezați pe structura de pixeli

Două reguli sunt luate în considerare: *repetarea texturii și filtrare*.

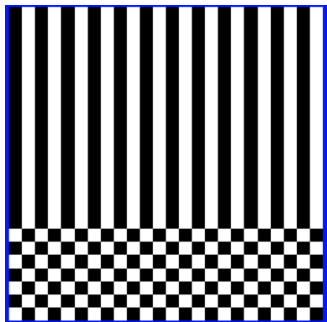
Ambele sunt legate de faptul că atât structura de texeli cât și cea de pixeli sunt discrete, iar scopul lor este de a indica modul în care se procedează atunci când nu există o corespondență 1:1 între cele două structuri. Forma generală a funcției asociate este

```
glTexParameter* (target, pname, value);
```

Parametrul `target` indică tipul de textură, în cazul 2D (considerat în continuare) acesta fiind `GL_TEXTURE_2D`. Ceilalți parametri depind ca semnificație și valoare de regula la care se face referire.

Repetarea texturii.

Motivație. Coordonatele de texturare, asociate inițial texturii, sunt reprezentate de pătratul $[0.0, 1.0] \times [0.0, 1.0]$. Cum se poate proceda în cazul în care sunt indicate valori în afara acestui pătrat? - detalii teoretice și exemple în secțiunea 3.



Repetarea texturii.

În acest caz (și ținând cont că suntem în context bidimensional) `pname` poate lua valorile

<code>GL_TEXTURE_WRAP_S</code>

<code>GL_TEXTURE_WRAP_T</code>

`s`, `t` sunt primele două coordonate ale texturii 2D, semnificația fiind că este precizată regula aplicată pentru coordonata indicată. În particular, pot fi utilizate reguli diferite pentru cele două coordonate de texturare.

Parametrul `value` poate avea una din valorile

<code>GL_CLAMP</code>

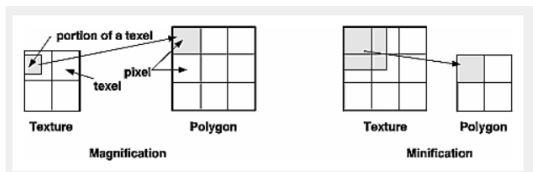
<code>GL_REPEAT</code>

În cazul lui `GL_CLAMP`, dacă se parcurge toată textura de-a lungul coordonatei considerate, valoarea ultimului texel este folosită în continuare pentru completarea tuturor pixelilor. În cazul lui `GL_REPEAT`, dacă se parcurge toată textura de-a lungul coordonatei considerate, se repornește parcurgerea texturii de la început, aceasta fiind repetată de câte ori este necesar.

Filtrare.

Motivație. Aici apare explicit faptul ca se poate întâmpla ca la randarea unei scene sa nu existe o corespondență biunivocă între texeli (elementele texturii) și pixeli (elementele buffer-ului de cadru).

Astfel, se poate ca un pixel sa corespundă unei porțiuni mici a structurii de texeli sau, invers, un pixel sa corespundă unei structuri de texeli, iar rolul regulii de filtrare este de a stabili cum anume structura de texeli este transferată pe cea de pixeli.



Sursa: *OpenGL RedBook*

Filtrare.

Parametrul `pname` poate avea, corespunzător celor două situații, valorile

<code>GL_TEXTURE_MAG_FILTER</code>	<code>GL_TEXTURE_MIN_FILTER</code>
------------------------------------	------------------------------------

Parametrul `value` poate avea una din valorile

<code>GL_NEAREST</code>	<code>GL_LINEAR</code>
-------------------------	------------------------

Semnificația valorilor este următoarea: pentru `GL_NEAREST` se alege texelul având coordonatele cele mai apropiate de centrul pixelului, în timp ce pentru `GL_LINEAR` este utilizată o tehnică de tipul *moving window*, fiind considerat un tablou de 2×2 texeli apropiați de centrul pixelului, după care se face o mediere.

Utilizarea propriu-zisă a texturii

În funcția de desenare/randare: textura trebuie activată / legată folosind funcțiile `glActiveTexture()` și `glBindTexture()`.

Ulterior, trebuie transmisă shader-ului de fragment ca variabilă de tip uniform - exemplu:

```
glUniform1i(glGetUniformLocation(ProgramId,"myTexture"),0) .
```

Comunicare cu shader-ele

Shader-ul de vârfuri: i se transmit, pe lângă coordonatele vârfurilor și culori, coordonatele de texturare (v. attributele); ca output sunt și poziția și culoarea și coordonatele de texturare.

Shader-ul de fragmente: are ca date de intrare atât informațiile transmise de shader-ul de vârfuri, cât și textura – folosind o variabilă uniformă (**uniform sampler2D**). Se poate folosi funcția `mix` pentru a “combina” culoarea sau diferite texturi.

Coordonate de modelare și coordonate de texturare

- Coordonatele de modelare sunt cele utilizate în mod curent în funcția de desenare. Fiecărui vârf îi sunt asociate, așa cum se știe, coordonatele spațiale $((x, y))$ în cazul scenelor 2D).

Coordonate de modelare și coordonate de texturare

- ▶ Coordonatele de modelare sunt cele utilizate în mod curent în funcția de desenare. Fiecărui vârf îi sunt asociate, așa cum se știe, coordonatele spațiale $((x, y)$ în cazul scenelor 2D).
- ▶ În cazul texturilor 2D coordonatele de texturare sunt s și t . Prin referire strict la **textură**, ambele sunt în intervalul $[0.0, 1.0]$. Cu alte cuvinte, coordonatele de texturare (teoretic) sunt reprezentate de pătratul $[0.0, 1.0] \times [0.0, 1.0]$.

Coordonate de modelare și coordonate de texturare

- ▶ Coordonatele de modelare sunt cele utilizate în mod curent în funcția de desenare. Fiecărui vârf îi sunt asociate, așa cum se știe, coordonatele spațiale $((x, y)$ în cazul scenelor 2D).
- ▶ În cazul texturilor 2D coordonatele de texturare sunt s și t . Prin referire strict la **textură**, ambele sunt în intervalul $[0.0, 1.0]$. Cu alte cuvinte, coordonatele de texturare (teoretic) sunt reprezentate de pătratul $[0.0, 1.0] \times [0.0, 1.0]$.
- ▶ În cadrul funcției de desenare, fiecărui vârf îi sunt asociate anumite coordonate de texturare. Când se realizează “legarea” de vârfuri pot fi considerate coordonate arbitrare, pe baza regulilor menționate anterior (v. slide-ul 15).

Coordonate de modelare și coordonate de texturare

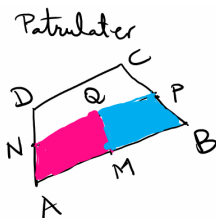
- ▶ Coordonatele de modelare sunt cele utilizate în mod curent în funcția de desenare. Fiecărui vârf îi sunt asociate, așa cum se știe, coordonatele spațiale $((x, y)$ în cazul scenelor 2D).
- ▶ În cazul texturilor 2D coordonatele de texturare sunt s și t . Prin referire strict la **textură**, ambele sunt în intervalul $[0.0, 1.0]$. Cu alte cuvinte, coordonatele de texturare (teoretic) sunt reprezentate de pătratul $[0.0, 1.0] \times [0.0, 1.0]$.
- ▶ În cadrul funcției de desenare, fiecărui vârf îi sunt asociate anumite coordonate de texturare. Când se realizează “legarea” de vârfuri pot fi considerate coordonate arbitrare, pe baza regulilor menționate anterior (v. slide-ul 15).
- ▶ Este suficient, pentru o primitivă grafică 2D, să fie indicate coordonatele de texturare pentru trei dintre vârfuri, coordonatele de texturare ale punctelor din interior rezultă automat. Motivație: dacă au fost fixate trei puncte necoliniare a, b, c în spațiul de texturare și trei puncte necoliniare A, B, C în spațiul de modelare, există o unică aplicație afină care transformă punctele a, b, c în A, B , respectiv C . Dacă sunt mai mult de trei vârfuri: coerență!

Despre aplicarea texturii

Coordonatele de texturare (teoretic) sunt reprezentate de pătratul $[0.0, 1.0] \times [0.0, 1.0]$. În cadrul funcției de desenare, fiecărui vârf îi sunt asociate anumite coordonate de texturare (valorile pot fi arbitrare, fiind aplicate regulile legate de repetare (slide 15)).

Despre aplicarea texturii


Este suficient, pentru o primitivă grafică 2D, să fie indicate coordonatele de texturare pentru trei dintre vârfuri, coordonatele de texturare ale punctelor din interior rezultă automat, folosind coliniaritatea și rapoarte de puncte coliniare. Dacă sunt mai mult de trei vârfuri: coerență!

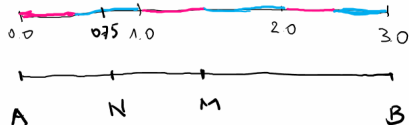


- Vî. A îi corresp. a , lui B... b , etc.
- Rapoartele din textură sunt "preluate" pt. patrulater

Textura $abcd$ este aplicată pe patrulaterul $ABCD$, modul de aplicare fiind dat de păstrarea coliniarității și a rapoartelor de puncte coliniare.

Un exemplu numeric

Se aplică o textură 1D  așa încât vârfurile $A = (30, 50)$ și $B = (70, 40)$ au coordonata de texturare 0.0, respectiv 3.0, opțiunea fiind GL_REPEAT. Fie N mijlocul lui $[AM]$, unde M este mijlocul lui $[AB]$. Ce culoare are N ?



Vârf	Coord. text.
A	0.0
B	3.0
M	1.5
N	0.75

\swarrow mijl. $[AB]$
 \swarrow mijl. $[AM]$

N are coord. de texturare 0.75,
 deci va fi reprezentat cu albastru


Concluzie

În concluzie, o textură 2D este:

- (i) un tablou dreptunghiular de texeli;
- (ii) o mulțime înzestrată cu coordonate de texturare (pătratul standard).

Atunci când tabloul de texeli are același număr de linii / coloane și atunci când textura este aplicată direct pe un pătrat, corespondența dintre pixeli și texeli este ușor de controlat și nu apar fenomene de deformare. În caz contrar (situație mai des întâlnită), trebuie manevrați în mod convenabil parametrii disponibili.

Exerciții

1. Se presupune că punctelor $(8.0, 7.0)$, $(6.0, 11.0)$, $(13.0, 13.0)$ din spațiul de modelare le sunt asociate coordonatele de texturare $(0.2, 0.4)$, $(0.6, 0.8)$, respectiv $(0.2, 0.2)$. Care sunt coordonatele de texturare ale punctului $(10.0, 11.0)$?
2. Se presupune că am generat o textură reprezentând o tablă de șah 8×8 și că aceasta este apelată folosind coordonatele de texturare $(0.0, 0.0)$, $(2.0, 0.0)$, $(2.0, 2.0)$, $(0.0, 2.0)$ și opțiunea `GL_REPEAT`. Câte pătrățele albe apar? (fondul este albastru)
3. Pe un fundal verde este desenat un pătrat folosind textura ; coordonatele de texturare asociate vârfurilor pătratului sunt $(0.0, 0.0)$, $(2.0, 0.0)$, $(2.0, 2.0)$, $(0.0, 2.0)$, iar opțiunea utilizată este `GL_CLAMP`. Care este raportul dintre suprafața colorată cu alb și cea colorată cu negru?