30/11/2015

FACULTATEA DE AUTOMATICA SI CALCULATOARE

ELEMENTE DE GRAFICA PE CALCULATOR

Tema 3

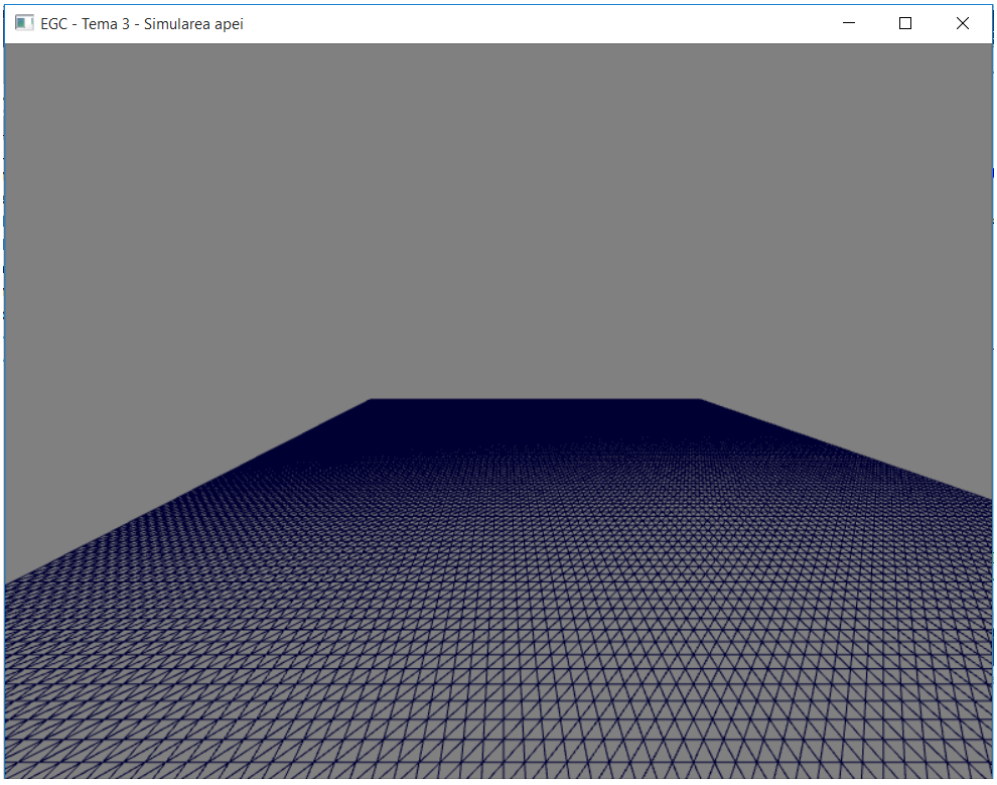
Simularea apei

Responsabil: Anca Morar
Termen: 15 decembrie 2015 ora 23:59

Nota: Orice nu este specificat in enunt este la latitudinea voastra. Daca aveti neclaritati puteti folosi forumul de la tema 3.

In cadrul acestei teme trebuie sa creati o plasa poligonala ce simuleaza apa si sa implementati iluminarea acesteia folosind modelul de shading Gouraud.

Va trebui sa creati (intr-un fisier sau programatic) o plasa poligonala ce defineste un grid. Un exemplu de astfel de grid este solul din laboratorul 6. Totusi, nivelul de detaliu al solului din laborator nu este destul de bun pentru cerintele temei. Plasa voastra poligonala trebuie sa aiba cel putin 100 x 100 varfuri.



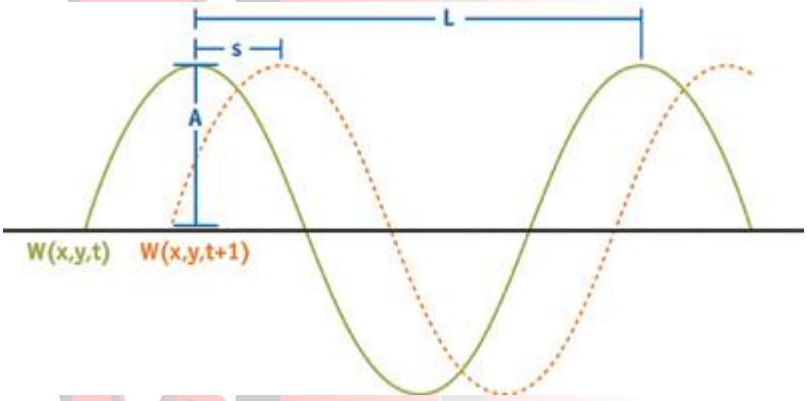
Daca se considera ca lucram intr-o lume in care coordonta y reprezinta inaltimea, pentru a face un teren tridimensional din gridul initial, fiecare varf din grid isi pastreaza coordonatele (x,z) dar isi modifica inaltimea (coordonata y).

Pentru simularea valurilor, se pot folosi diferite modele de deformare a gridului. O metoda posibila de realizare a acestei deformari este insumarea unor curbe de tip sinus cu diferiti parametri, pentru calculul inaltimii:

$$y(x,z,t) = 2 \times A_i \times \left(\frac{\sin(D_i \cdot (x,z) \times w_i + t \times \varphi_i) + 1}{2} \right)^2$$

unde

- A_i este amplitudinea undei
- $D_i(x,z)$ este directia undei
- w_i este frecventa undei - are legatura cu lungimea de unda L astfel: $w = 2\pi/L$
- φ_i este faza - are legatura cu viteza S (distanta cu care creasta valului se deplaseaza per secunda): $\varphi = S \times 2\pi/L$
- x este produs normal (nu vectorial), iar $\cdot$ este produs scalar



Daca undele sunt directionale, D_i este constanta pe intreaga durata de viata a undei. Daca undele sunt circulare, D_i trebuie calculata pentru fiecare varf:

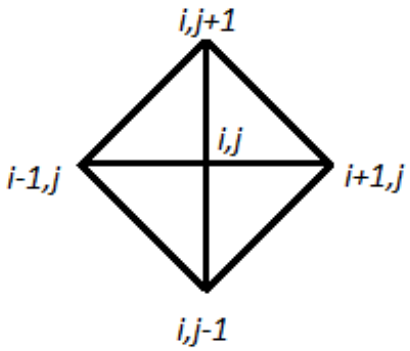
$$D_i(x,z) = \frac{C_i - (x,z)}{|Ci - (x,z)|}$$

unde $C_i(x,z)$ este centrul udei, iar (x,z) sunt coordonatele x si z ale varfului.

Pentru mai multe explicatii privind formula prezentata, puteti citi articolul urmator: http://http.developer.nvidia.com/GPUGems/gpugems_ch01.html

Initial toate varfurile plasei poligonale ar trebui sa aiba normala (0,1,0). Dupa deplasarea varfurilor, aceste normale se modifica si ele, deci, pentru o iluminare corecta, normalele trebuie recalculate. Pentru calculul normalelor se poate alege solutia bazata pe derivate (ca in articol), sau se poate aproxima in modul urmator:

- Deoarece in vertex shader aveti prelucrarile pentru varful curent, si nu aveti acces la pozitiile altor varfuri, trebuie sa calculati unde sunt pozitionate varfurile vecine varfului curent. Astfel, pentru varful curent, se calculeaza cinci inaltimi (cu formula precendenta):
    - Una in varful curent V(i,j). Rezulta coordonatele varfului V_1
    - Una in varful predecesor pe axa X, V(i-1,j). Rezulta varful V_2
    - Una in varful urmator pe axa X, V(i+1,j). Rezulta varful V_3
    - Una in varful predecesor pe axa Z, V(i,j-1). Rezulta varful V_4
    - Una in varful urmator pe axa Z, V(i,j+1). Rezulta varful V_5
- Unind V_1 cu celelalte varfuri obtinem patru muchii cu care putem defini patru triunghiuri, ca in figura de mai jos. Pentru fiecare din cele patru triunghiuri se calculeaza normala per triunghi (prin produs vectorial intre doua laturi ale triunghiului). Media celor patru normale obtinute este normala varfului curent. Acum daca aveti normala in fiecare varf, puteti calcula corect iluminarea (ca in laboratorul 6).

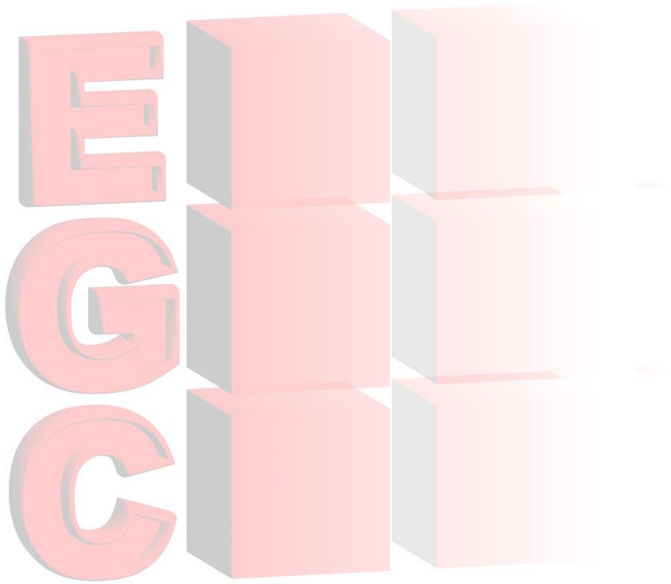


Cerinte:

- Crearea plasei poligonale nedeformate (citire din fisier sau programatic), folosind vao, vbo, ibo si trimiterea la vertex shader a pozitiilor varfurilor.
- Stabilirea numarului de sinusoide care afecteaza plasa poligonala (cel putin 2) si a parametrilor acestora: A_i, D_i (x,z), $C_i(x,z)$, w_i si φ_i (pe CPU si trimisi la vertex shader). Tot pe CPU se stabileste care dintre unde sunt circulare si care sunt directionale (si se transmite mai departe la vertex shader).
- Calculul undei in vertex shader folosind procesul prezentat anterior (sau unul mai avansat): atat pozitiile finale ale varfurilor cat si normalele acestora **se vor calcula**

in vertex shader (nu se transmit de la CPU). Se vor implementa atat unde directionale, cat si circulare

- Iluminare folosind modelul de shading Gouraud (calculul culorii varfurilor in vertex shader) si modelul de reflexie care sa contina si componenta speculara (Phong/Blinn-Phong/etc.)
- Lumina cu care este iluminata apa se poate deplasa cu tastele
- Implementarea unei camere FPS complet functionale (si translatie si rotatie), cu care sa se poata explora spatiul



○ BAREM ORIENTATIV

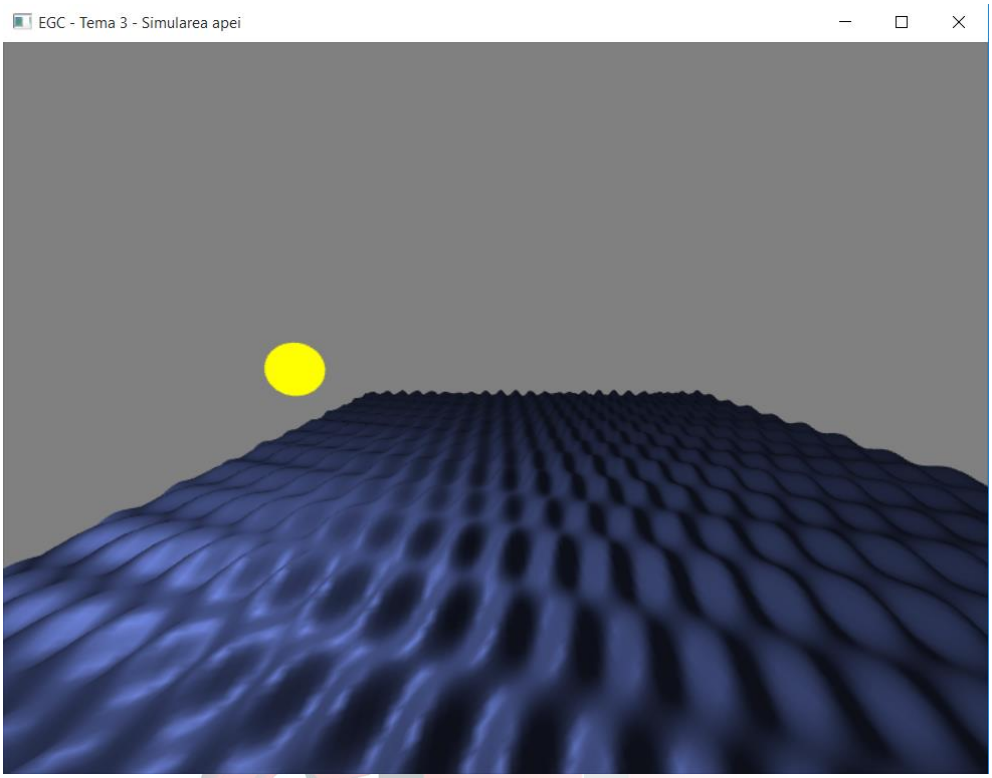
- Crearea plasei poligonale initiale 10%
- Calculul varfurilor din plasa poligonala deformata 30%
- Calculul normalelor varfurilor in plasa deformata 30%
- Iluminarea folosind modelul de shading Gouraud 20% (o lumina care poate fi deplasata cu tastele)
- Camera FPS functionala 10%

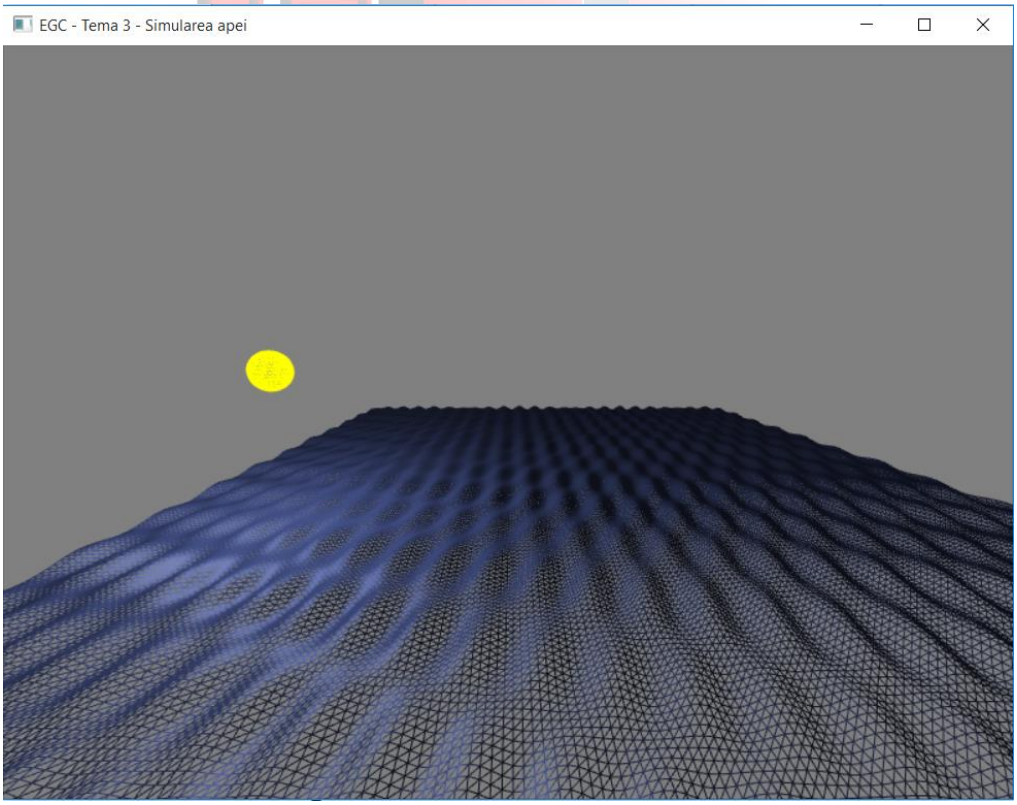
BONUSURI

- Alte metode de generare a undelor, mai avansate decat cea explicata aici (de exemplu: Gerstner, Stokes-Navier)
- Iluminare in fragment shader (Phong shading) – nu este un bonus foarte mare deoarece se va implementa si la laborator saptamanile viitoare (inainte de deadline-ul temei)
- Atenuarea valurilor in timp
- Orice aduce realism scenei

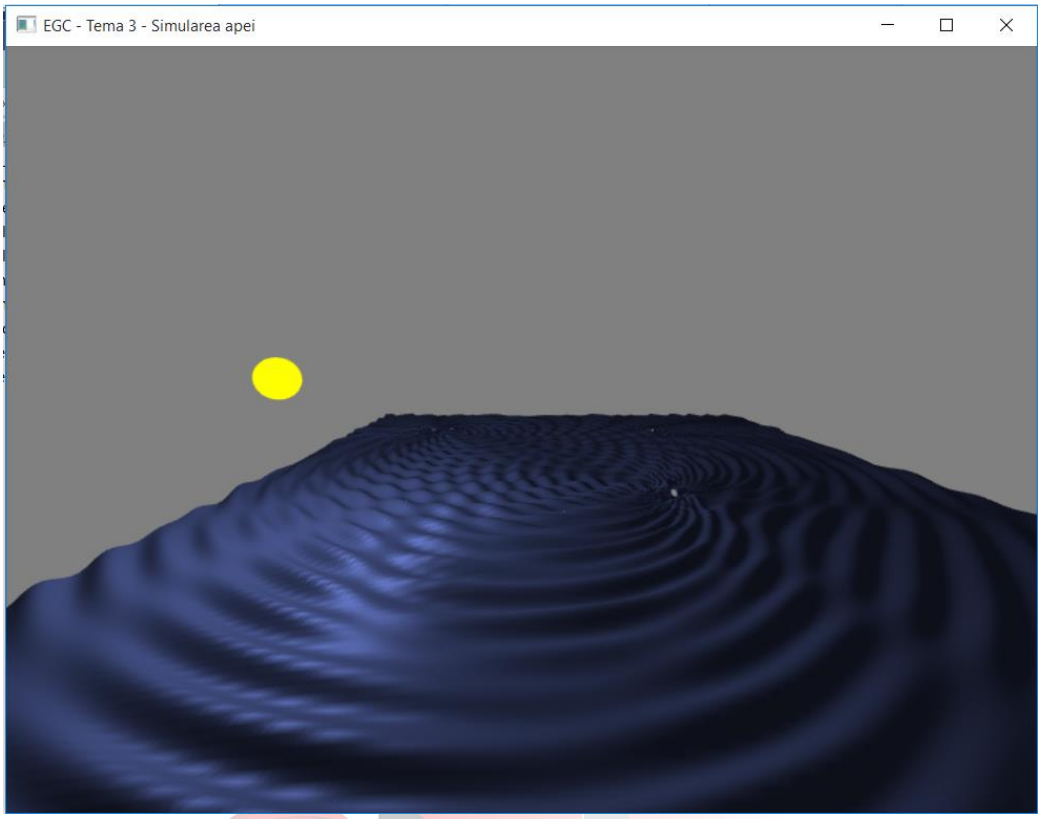
ANEXE

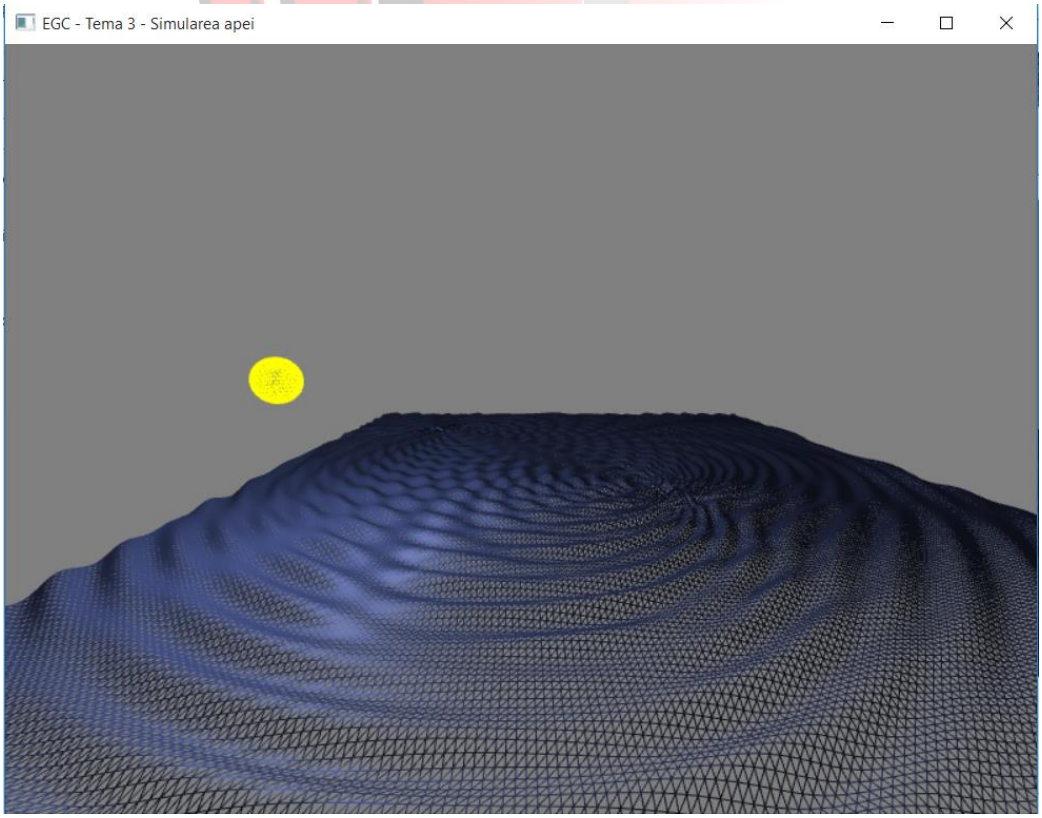
Exemplu de plasa deformata de 3 unde directionale:





Exemplu de plasa deformata de trei unde circulare:





Exemplu de plasa deformata de o unda directionala si una circulara: