14/12/2012

FACULTATEA DE AUTOMATICA SI CALCULATOARE

ELEMENTE DE GRAFICA PE CALCULATOR

Laborator 8

Stencil Buffer-ul

Stencil buffer-ul pune la dispozitie multe optiuni pentru a restrictiona desenarea pe ecran si se poate folosi in multe locuri acolo unde bufferul de adancime nu poate fi utilizat. Privit in modul cel mai simplist stencil buffer-ul este folosit pentru a bloca la desenare anumite zone din ecran.

Poate cea mai spectaculoasa folosire a stencil bufferului este la desenarea de umbre. In functie de hardware-ul grafic se pot genera umbre hard sau soft folosind una sau mai multe surse de lumina.

Utilizare

Activarea stencil buffer-ului se face folosind *glEnable(GL_STENCIL_TEST)*. Fara acest apel toate operatiile stencil sunt dezactivate.

Exista 4 functii stencil in OpenGL :

- *void glClearStencil(GLint s)*
- *void glStencilFunc(GLenum func, GLint ref, GLuint mask)*
- *void glStencilMask(GLuint mask)*
- *void glStencilOp(GLenum fail, GLenum zfail, GLenum zpass)*

glClearStencil() functioneaza in mod similar cu glClearColor si glClearDepth; deasemenea initializeaza stencil buffer-ul cu valoarea initiala data ca parametru cand este apelat glClear(GL_STENCIL_BIT). In mod default aceasta valoare este 0.
In mod default doar prin activarea folosirii stencil bufferului nu se intampla nimic toate desenarile pe ecran fiind permise fara nici o restrictie. Pentru a lucra cu stencil bufferul trebuie sa "marcam" valori in acesta folosind glStencilFunc() si glStencilOp().

Functia *glStencilFunc()* defineste :

- o functie de comparatie
- o valoare de referinta cu care se face comparatia
- o masca (se face SI logic cu valoarea de referinta)

Functiile de comparare valide sunt :

- GL_NEVER - stencil test-ul esueaza intotdeauna (nu se va desena nimic pe ecran)
- GL_ALWAYS - stencil test-ul reuseste intotdeauna (desenarea are loc mereu - aceasta este comportarea default)
- GL_EQUAL - stencil test-ul reuseste daca valoarea de referinta este egala cu valoarea din stencil buffer
- GL_NOTEQUAL - stencil test-ul reuseste daca valoarea de referinta nu este egala cu valoarea din stencil buffer

- GL_LESS - stencil test-ul reuseste daca valoarea de referinta este mai mica decat valoarea din stencil buffer
- GL_LEQUAL - stencil test-ul reuseste daca valoarea de referinta este mai mica sau egala cu valoarea din stencil buffer
- GL_GREATER - stencil test-ul reuseste daca valoarea de referinta este mai mare decat valoarea din stencil buffer
- GL_GEQUAL - stencil test-ul reuseste daca valoarea de referinta este mai mare sau egala cu valoarea din stencil buffer

Legate de functiile stencil sunt operatiile stencil definite cu glStencilOp care seteaza comportarea valorii stencil bufferului pentru cei trei parametrii ai functiei :

- fail : testul stencil a esuat
- zfail : testul stencil a reusit dar testul de adancime a esuat
- zpass : testul stencil a reusit si testul de adancime a reusit

Operatiile valide asupra valorilor din stencil buffer sunt :

- GL_ZERO - seteaza valoare din stencil pentru pixelul/fragmentul curent la 0
- GL_KEEP - pastreaza valoarea din stencil pentru pixelul/fragmentul curent
- GL_REPLACE - inlocuieste valoarea din stencil pentru pixelul/fragmentul curent cu valoarea de referinta
- GL_INCR - mareste valoarea din stencil pentru pixelul/fragmentul curent
- GL_DECR - scade valoarea din stencil pentru pixelul/fragmentul curent

Exemple de folosire :

- glStencilFunc(GL_ALWAYS, 1, 1);
  glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);
  Se va desena tot timpul pe ecran iar in locul unde a fost desenat ceva stencil bufferul va avea valoarea 1
- glStencilFunc(GL_EQUAL, 0, 1);
  glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);
  Se va desena pe ecran numai unde stencil bufferul este 0 si valorile vor fi pastrate

Projective Shadows

O metoda simpla care permite afisarea de umbre pe suprafete plane a fost dezvoltata in 1988 de James Blinn.
Ideea care sta in spatele metodei propuse de Blinn este aceea de a gasi o transformare care proiecteaza geometria feicarui obiect generator de umbre (occluder) pe un plan stabilit rezultand astfel umbra obiectului respectiv.
Proiectia poate fi descrisa de o matrice 4x4 de transformare in coordonate omogene.

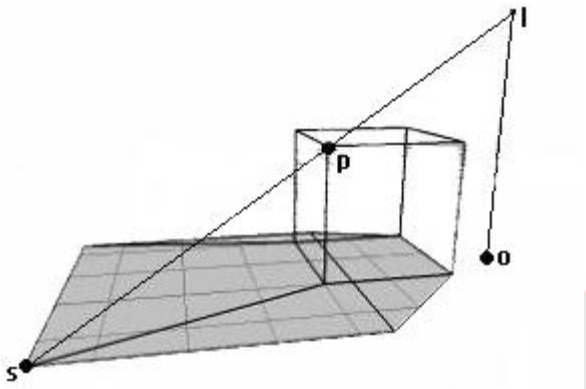


Fig1: Umbra proiectata folosind o sursa de lumina punctiforma

Pentru a simplifica lucrurile sa consideram pentru inceput situatia din figura unde avem o sursa de lumina la pozitia **L** = (Lx,Ly,Lz) si planul care va primi umbra este y = 0
Un punct **P** = (Px,Py,Pz) de pe obiect vazut din sursa de lumina **L** va proiecta o umbra pe plan in punctul **S** = (Sx,0,Sz).
Punctul **S** poate fi exprimat ca :

$$s = l + t(p - l). \qquad (1)$$

Dar cum Sy = 0 rezulta :

$$t = \frac{l_y}{l_y - p_y}$$

Introducand in formula (1) obtinem coordonatele lui **S** :

$$\begin{pmatrix} s_x \\ 0 \\ s_z \\ 1 \end{pmatrix} = \begin{bmatrix} l_y & -l_x & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -l_z & l_y & 0 \\ 0 & -1 & 0 & l_y \end{bmatrix} \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

In cazul general poligonul pe care se vor proiecta umbrele nu se afla in planul y = 0. Pentru un plan oarecare **NX** + d = 0, unde **N** este vectorul normalei planului si **X** este un punct oarecare de pe plan se poate construi matricea transformarii **M** intr-un mod similar. Combinand formula (1) cu ecuatia planului si introducand pe **X** = **S** rezulta :

$$t = \frac{\mathbf{nl} + d}{\mathbf{n} \cdot (\mathbf{l} - \mathbf{p})}$$

Similar rezulta matricea de transformare **M** care satisface relatia **S** = **MP**

$$\mathbf{M} = \begin{bmatrix} \mathbf{nl} + d - l_x n_x & -l_x n_y & -l_x n_z & -l_x d \\ -l_y n_x & \mathbf{nl} + d - l_y n_y & -l_y n_z & -l_y d \\ -l_z n_x & -l_z n_y & \mathbf{nl} + d - l_z n_z & -l_z d \\ -n_x & -n_y & -n_z & \mathbf{nl} \end{bmatrix}$$

Observatii

Faptul ca metoda propusa de Blinn este foarte simpla duce la o serie de probleme/limitari :

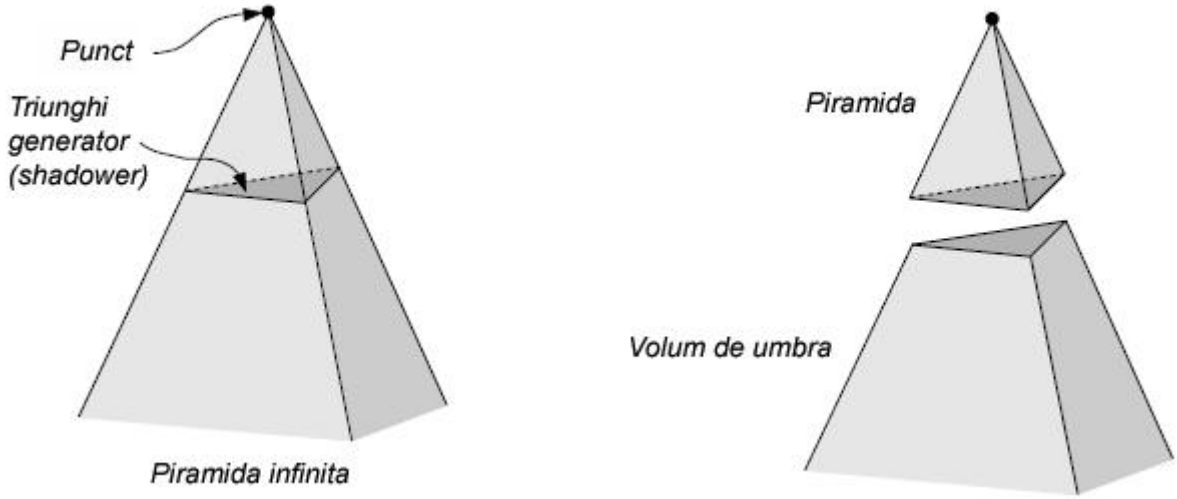
- limitarea la suprafete plane
- datorita problemelor de precizie este posibil ca poligoanele proiectate sa fie afisate sub cele ale planului pe care se proiecteaza umbrele
- o alta problema este data de faptul ca umbrele sunt proiectate pe planul ce contine poligonul primitor si nu direct pe acesta; astfel umbra va fi desenata si in afara poligonului (puteti observa acest lucru foarte bine si in codul sursa pentru acest laborator); acest neajuns poate fi rezolvat prin folosirea stencil bufferului (bufferul de marcaj)
- urmatoarea problema este data de faptul ca vor fi afisate umbre chiar si pentru obiectele care nu pot in mod normal sa aiba umbre; daca obiectul este situat in spatele sursei de lumina umbrele generate de acesta se numesc anti-umbre (anti-shadows) iar daca obeictul se afla in spatele planului umbrele generate se numesc umbre false (fake shadows); o solutie pentru eliminarea acestor probleme ar fi folosirea de clipping-planes

Avantaje/Dezavantaje

- Avantaje
  - realizeaza proiectarea umbrelor pe suprafete plane cu rezultate f. bune
  - metoda rapida pentru scene mici cu o singura sursa de lumina si un plan pe care se proiecteaza umbrele
- Dezavantaje
  - nu permite obiectelor concave sa se auto-umbreasca
  - obiectele nu se umbresc unul pe celalalt
  - creaza hard shadows
  - daca se doreste realizarea de umbre pentru surse de lumina multiple si mai multe plane costul computational creste foarte repede

Shadow Volumes folosind stencil buffer-ul

Folosind aceasta metoda se pot proiecta umbre pe obiecte arbitrare si se poate realiza si auto-umbrirea obiectelor
Pentru inceput sa ne imaginam o sursa de lumina punctiforma si un poligon generator (shadower). Luand ca poligon generator un triunghi extinzand virtual liniile din sursa de lumina si care trec prin varfurile tringhiului obtinem o piramida infinita. Putem considera partea de deasupra ca fiind o piramida iar partea de sub poligon ca fiind un trunchi de piramida. Orice parte a unui obiect care se afla in interiorul volumului trunchiului de piramida va fi in umbra. Denumirea acestui volum este "volum de umbra" (shadow volume).



Sa ne imaginam ca vizualizam scena dintr-un anumit punct din spatiu iar pentru fiecare pixel afisat ducem o raza pana cand aceasta va lovi obiectul afisat pe ecran. Pe masura ce raza este proiectata spre obiect incrementam/decrementam un contor de fiecare data cand aceasta intra in volumul de umbra (traverseaza o fata front-facing a acestuia) si decrementam/incrementam acelasi contor cand raza iese din volumul de umbra (traverseaza o fata back-facing a acestuia).

Daca in momentul in care raza a lovit pixelui obiectului contorul este mai mare decat 0 atunci acel pixel se afla in volumul de umbra (este umbrit) iar in caz contrar se afla in afara acestuia. Daca aceasta metoda ar fi aplicata geometric pentru fiecare pixel al obiectului ar necesita un numar foarte mare de calcule. De aceea in practica se foloseste stencil-bufferul care va face contorizarea.

Folosind si stencil-bufferul si z-Bufferul se deseneaza pe rand intai fetele front-facing ale volumului de umbra avand grija sa incrementam/decrementam contorul din stencil-buffer iar apoi sunt desenate fetele back-facing ale volumului de umbra avand grija sa decrementam/incrementam contorul.

In final este afisata intreaga scena, o valoare de 0 in stencil-buffer insemanand ca acel pixel se afla in afara volumului de umbra iar o valoare de 1 inseamna ca acesta este umbrit.

Pe scurt iata algoritmul folosit :

- 1. Bufferul de culoare si adancime sunt activate pentru scriere si testul de adancime este activat
- 2. Se seteaza atributele pentru desenul umbrei si dezactiveaza sursa de lumina
- 3. Este desenata scena
- 4. Se calculeaza poligoanele care genereaza volumul de umbra
- 5. Se dezactiveaza bufferele de culoare si adancime referitor la scriere (acestea nu mai primesc updateuri) dar testul de adancime ramane activat
- 6. Se sterge stencil bufferul cu valoarea 0 daca observatorul se afla in afara volumului de umbra sau cu 1 in caz contrar
- 7. Se seteaza functia stencil sa permita intotdeauna afisarea (GL_ALWAYS)
- 8. Se seteaza operatia stencil sa decrementeze bufferul de marcaj daca testul de adancime reuseste
- 9. Se activeaza back-face cullingul (vor fi desenate doar fetele front-facing)
- 10. Se deseneaza poligoanele volumului de umbra
- 11. Se seteaza operatia stencil sa incrementeze bufferul de marcaj daca testul de adancime reuseste
- 12. Se activeaza front-face cullingul (vor fi desenate doar fetele back-facing)
- 13. Se deseneaza poligoanele volumului de umbra
- 14. Se seteaza functia stencil sa testeze egalitatea cu 0 (desenarea a ceea ce se afla in afara volumului de umbra)
- 15. Se seteaza operatiile stencil sa nu faca nimic si sa pastreze stencil bufferul la valoarea actuala (GL_KEEP)
- 16. Se activeaza sursa de lumina
- 17. Se deseneaza scena
- Optional daca se doreste afisare partilor umbrite cu o alta culoare (de exemplu un negru mai deschis) se pot realiza urmatorii pasi suplimentari :
  - 18. Se seteaza functia stencil sa testeze egalitatea cu 1 (desenarea a ceea ce se afla in volumul de umbra)
  - 19. Se dezactiveaza sursa de lumina
  - 20. Se deseneaza scena