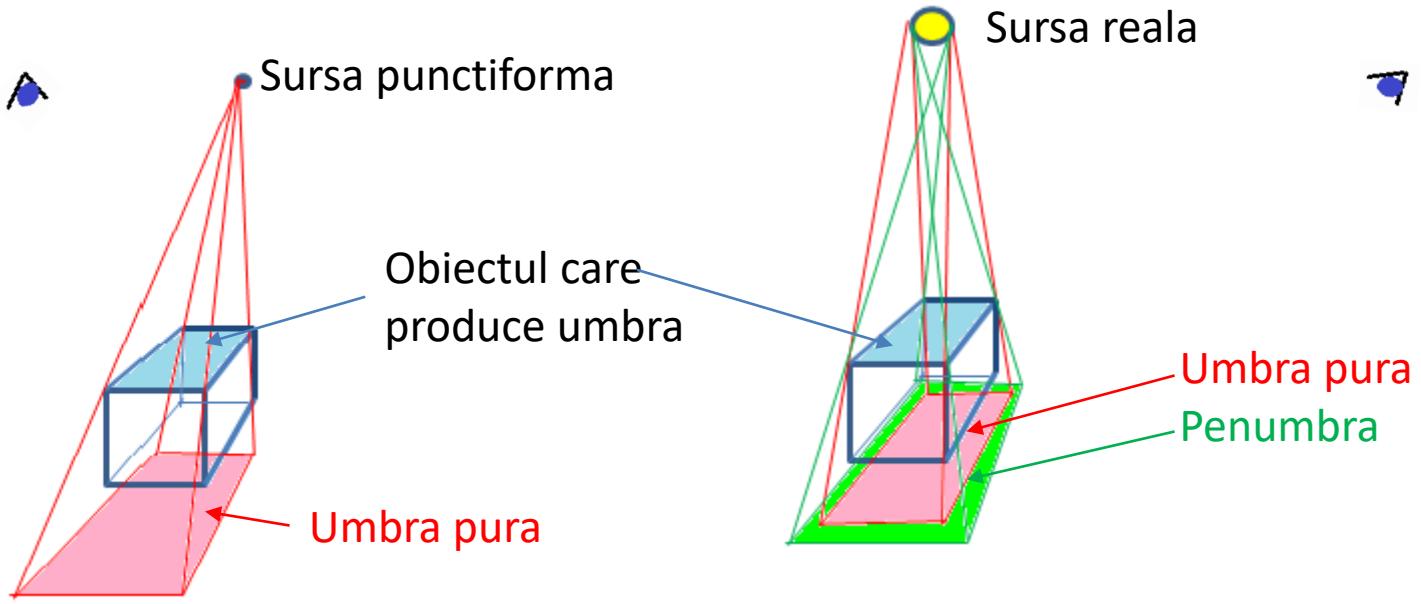
Redarea umbrelor în imagini -1

Prof. univ. dr. ing. Florica Moldoveanu

Curs *Elemente de Grafică pe Calculator* – UPB, Automatică şi Calculatoare
2021-2022

INTRODUCEREA UMBRELOR IN IMAGINI(1)

- Atunci când un observator priveşte o scenă 3D luminată de o sursă de lumină dintr-o poziţie diferită de aceea a sursei de lumină, va vedea umbrele produse de obiectele scenei.
- Umbrele au o contribuţie însemnată la realismul imaginii, îmbunătăţind percepţia profunzimii.



In general, se considera numai umbra pura: calcule mai simple.

INTRODUCEREA UMBRELOR IN IMAGINI(2)

❑ **Volumul de calcule necesar introducerii umbrelor depinde de poziţia sursei de lumină**:

▪ Dacă sursa este la infinit calculele sunt mai simple: se efectueaza o proiectie paralela din pozitia sursei.

▪ In cazul unei surse situate la distanţă finită dar în afara scenei, este necesară o proiecţie perspectivă din poziţia sursei.

▪ Cazul cel mai dificil este acela în care sursa este situată în scenă.

❑ Problema determinării umbrelor este similară aceleia de determinare a părţilor nevizibile ale unei scene 3D: partile scenei 3D care nu sunt vizibile din poziţia sursei de lumină sunt în umbră.

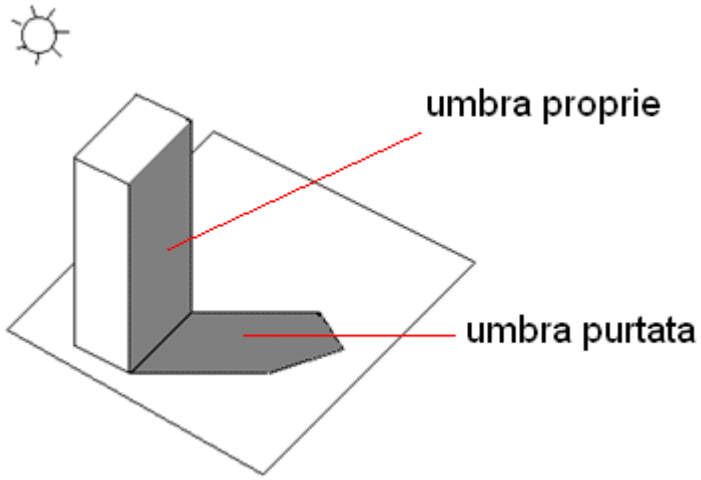
➢ Crearea unei imagini cu umbre presupune rezolvarea de două ori a problemei suprafeţelor nevizibile:

- privind scena din poziţia sursei de lumină – determinare parti umbrite
- Privind scena din poziţia observatorului – determinare parti vizibile in imagine

UMBRE PLANARE (1)

- Umbrele produse de obiectele scenei 3D sunt de două tipuri:

  umbre proprii şi **umbre purtate**.



– **Umbrele proprii** sunt generate de obiectul însuşi, care împiedică lumina să ajungă la unele dintre feţele sale: feţele umbrite de umbra proprie sunt feţele auto-obturate atunci când scena este văzută din poziţia sursei de lumină.

– **Umbrele purtate** sunt umbrele pe care un obiect le produce pe alte obiecte ale scenei, la care lumina nu ajunge din cauza obiectului.

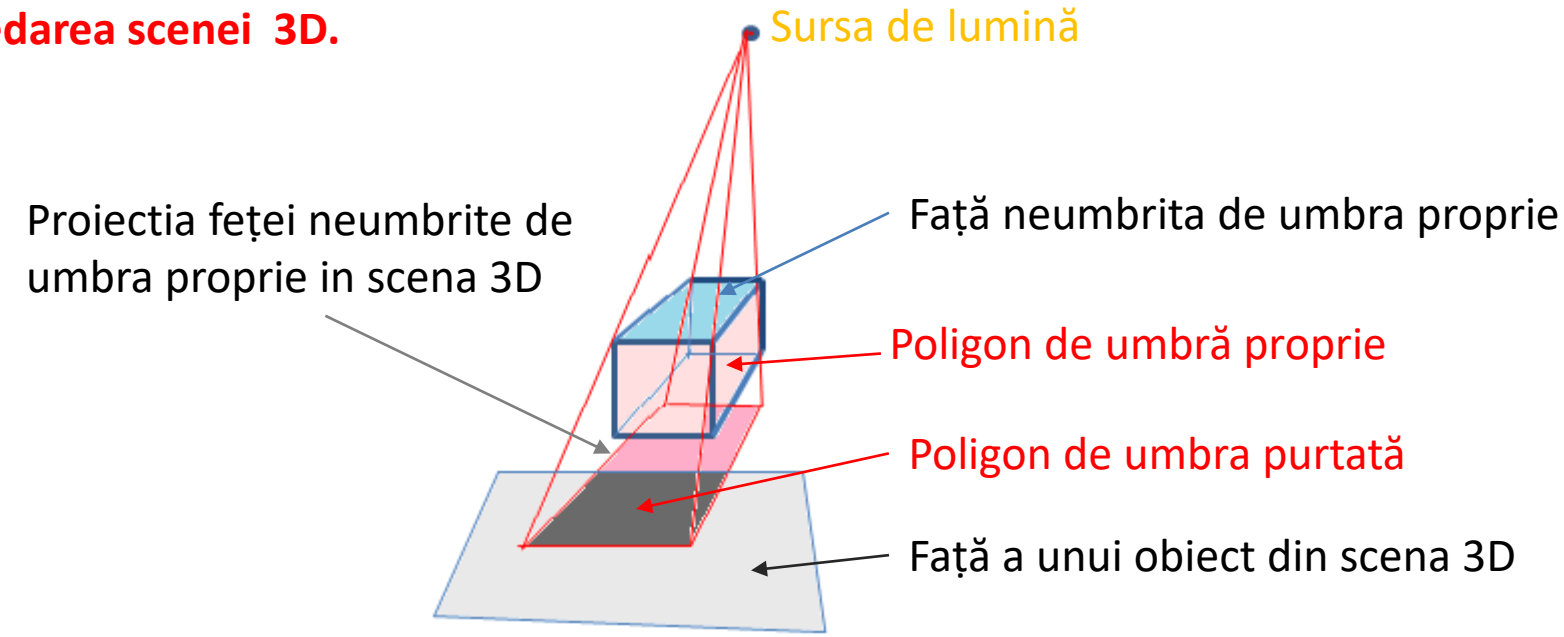
UMBRE PLANARE (2)

❑ Umbrele purtate se pot determina proiectând în scenă, din poziţia sursei, toate poligoanele (feţele) neumbrite de umbra proprie. Proiectiile acestor poligoane pe obiectele scenei 3D se numesc **poligoane de umbră purtată.**

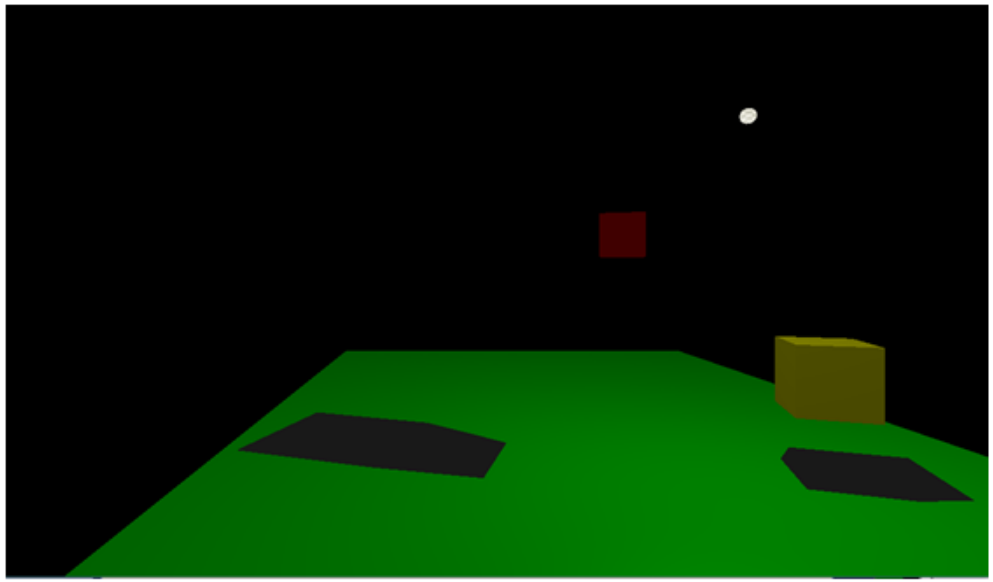
❑ **Un poligon de umbră purtată este coplanar cu fața obiectului care contine umbra purtată.**

❑ **Un poligon de umbră proprie este un poligon identic cu fața umbrită de umbra proprie.**

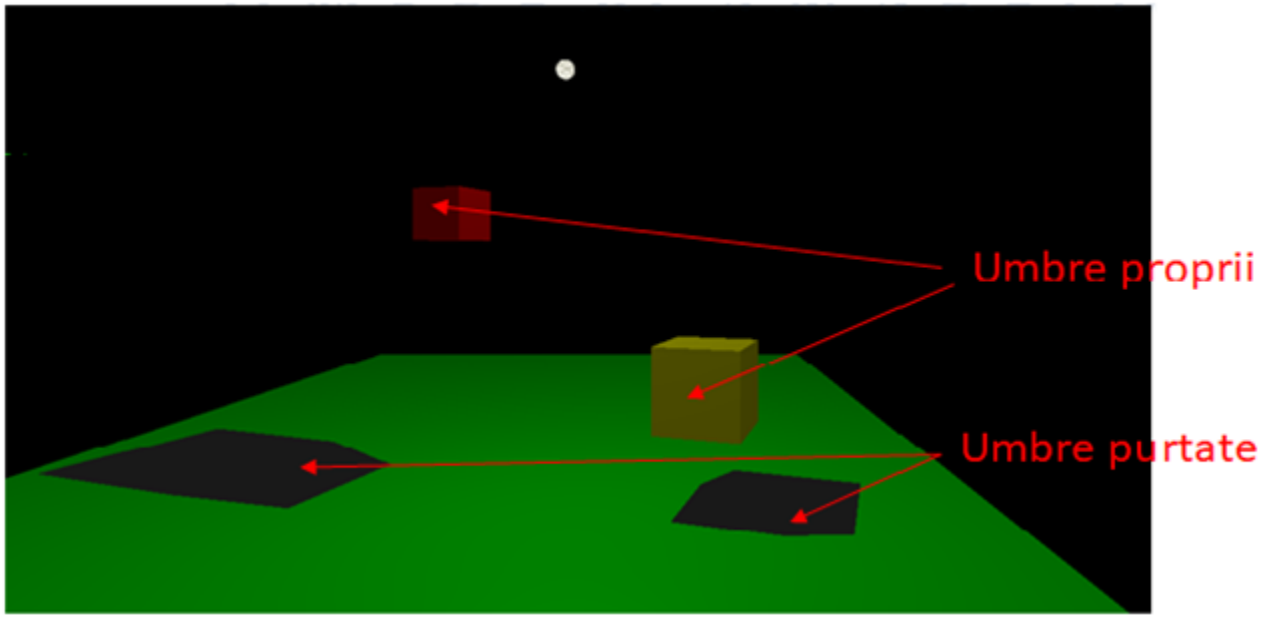
❑ **Poligoanele de umbră proprie şi cele de umbra purtată se folosesc ca poligoane detaliu la redarea scenei 3D.**



UMBRE PLANARE (3)



Scena cu umbre vazuta dintr-o pozitie a observatorului diferita de cea a sursei.

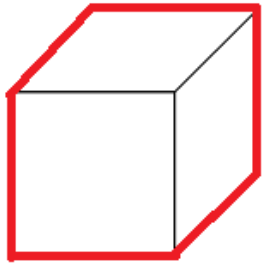


UMBRE PLANARE (4)

Redarea scenei cu umbre (functia Update)

1. Se cere rasterizarea poligoanelor scenei.
2. Se dezactivează testul de adâncime (astfel incat fragmentele poligoanelor de umbra să suprascrie unele dintre fragmentele poligoanelor scenei): glDisable(GL_DEPTH_TEST).
3. Se cere rasterizarea poligoanelor de umbră (acestea au o culoare de umbra)

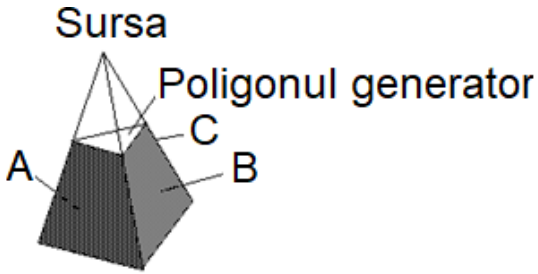
- Pot fi generate mai multe vederi ale scenei fără a recalcula umbrele, deoarece umbrele depind numai de poziţia sursei (surselor) de lumină.
- Numărul de poligoane de umbră este mai mic, dacă în loc să se proiecteze fiecare faţă luminată de sursă, se proiectează silueta fiecărui obiect văzută din poziţia sursei de lumină.



- Metoda are mai multe dezavantaje (umbrele pot fi redate numai pe suprafete plane, complexitatea calculelor creste rapid cu numarul surselor de lumina, ş.a), de aceea in prezent se folosesc alte metode pentru introducerea umbrelor in imagini.

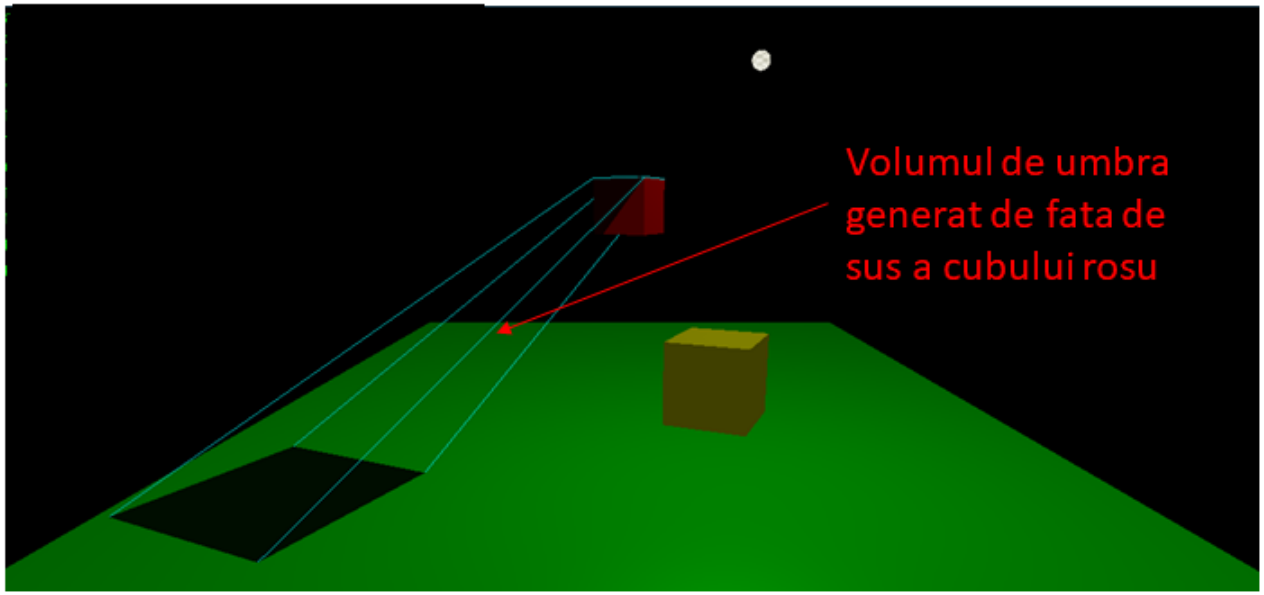
VOLUME DE UMBRA (1)

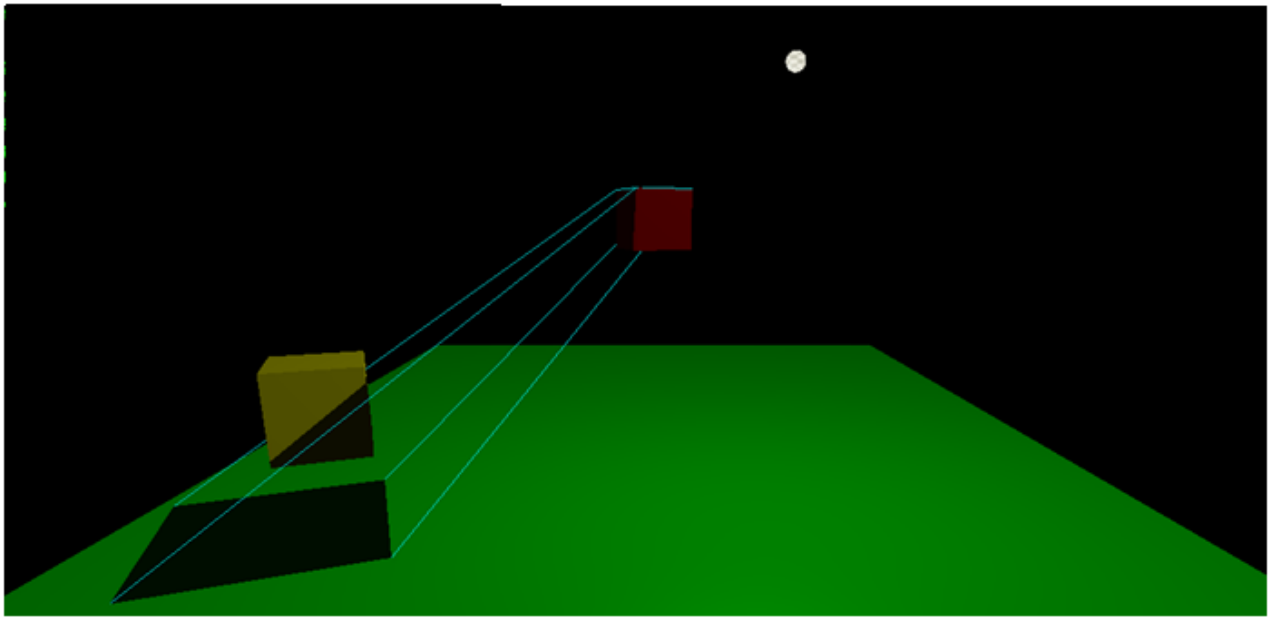
- Sursa de lumină este considerată punctiformă iar scena 3D alcatuita din poligoane.
- Un **volum de umbră** este definit de o sursă de lumină şi un poligon luminat (vizibil din poziţia sursei de lumină), pe care-l vom numi **poligonul generator**.



- Volumul de umbra determinat de o sursă şi un poligon generator este delimitat de o faţă care reprezintă poligonul generator scalat. Această faţă este situată la o distanţă faţă de sursă dincolo de care intensitatea luminii sursei este neglijabilă, deci orice punct aflat dincolo de această limită este umbrit.
- Volumul de umbră este decupat la marginile volumului vizual.

VOLUME DE UMBRA (2)

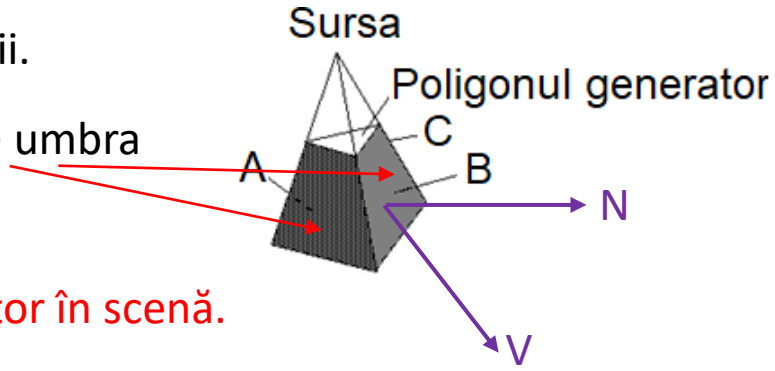




VOLUME DE UMBRA (3)

Determinarea umbririi produse in scena 3D de un volum de umbra

❑ Feţele laterale ale volumului sunt numite **poligoane de umbră**. Un poligon de umbră este determinat de o latură a poligonului generator şi cele două drepte care pleacă din sursa de lumină, fiecare trecând printr-un vârf al laturii.



❑ Poligoanele de umbră se folosesc pentru determinarea umbririi produse de poligonul generator în scenă.

❑ Normalele la poligoanele de umbra trebuie sa fie orientate spre exteriorul volumului: poligoanele de umbra sunt orientate trigonometric atunci cand sunt vazute din exteriorul volumului de umbra.

– PUV poligoanele de umbră care sunt vizibile din poziţia observatorului (A şi B în figura) şi cu

– PUN poligoanele de umbră care nu sunt vizibile din poziţia observatorului (poligonul C).

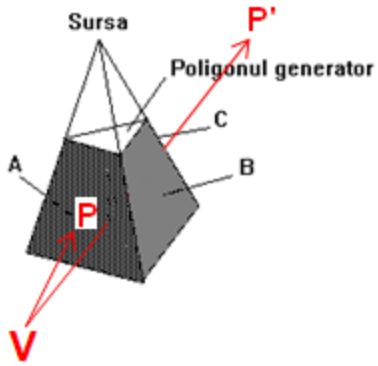
➢ Clasificarea poligoanelor de umbra in PUV si PUN se poate face pe baza normalelor la poligoane si a pozitiei observatorului (back face culling).

VOLUME DE UMBRA (4)

Fie

– un punct P al unei suprafeţe din scena 3D şi

– VP vectorul din poziţia observatorului (V) în punctul P.



Punctul P este umbrit dacă numărul de poligoane de tip PUV intersectate de vectorul VP este mai mare decât numărul de poligoane de tip PUN intersectate de vector.

Acesta este singurul caz, atunci când punctul V nu este în volumul de umbră.

❑ In general, pentru a determina dacă un punct P este în umbră, se poate folosi un contor în care iniţial se memorează numărul de volume de umbră care conţin poziţia observatorului.

– Se asociază poligoanelor de tip PUV valoarea +1 iar celor de tip PUN valoarea -1.

– Atunci când vectorul VP traversează un poligon de umbră, se adună la contor valoarea asociată poligonului.

– Punctul P este umbrit dacă valoarea contorului este pozitivă în P.

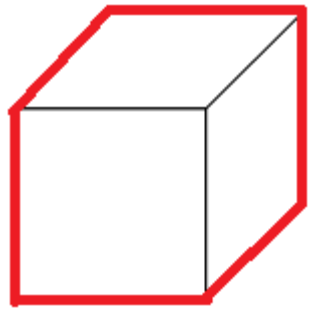
VOLUME DE UMBRA (5)

➢ Volumul de calcule necesar acestui algoritm poate fi redus dacă în loc să se calculeze volumul de umbră pentru fiecare poligon vizibil din poziţia sursei, se calculează un singur volum de umbră pentru o suprafaţă poliedrală.

In acest scop, se determină poligoanele de umbră numai pentru laturile care fac parte din silueta suprafeţei, văzută din poziţia sursei de lumina.

❑ Silueta unei suprafeţe, corespunzătoare unui punct de observare, este un set conectat de laturi care aparţin poligoanelor vizibile din punctul de observare.

❑ O latură de siluetă este fie o latură de margine a unei suprafeţe deschise, fie o latură care separă un poligon vizibil de unul nevizibil.



➢ Pentru determinarea laturilor de siluetă, este necesar să se folosească o structură de date care reflectă adiacenţa poligoanelor. Cunoscându-se poligonul adiacent pe fiecare latură a fiecărui poligon vizibil din poziţia sursei, se pot determina rapid laturile de siluetă.

VOLUME DE UMBRA (6)

Implementarea metodei folosind stencil-buffer(1)

- Buffer-ul de marcaje (stencil-buffer) este prezent pe placile grafice actuale (in memoria GPU), alaturi de buffer-ul imagine (frame-buffer) si buffer-ul de adancime (z-buffer).
  - este o memorie de numere intregi asociata pixelilor imaginii, de regula un byte/pixel;
  - poate fi utilizat pentru a implementa diferite operatii grafice, de regula in combinatie cu z-buffer, cum este si cazul implementarii volumelor de umbra:
  - valorile din stencil-buffer pot fi incrementate/ decrementate automat pentru fiecare fragment care trece/ nu trece testul de adancime.

- Exista mai multe metode de a utiliza stencil-buffer pentru implementarea metodei volumelor de umbre.

VOLUME DE UMBRA (7)

Implementarea metodei folosind stencil-buffer(2)

- In metoda prezentata in continuare, se creaza in stencil-buffer o masca a umbrelor din imagine: orice pixel de umbra, (x,y), va fi reprezentat in stencil-buffer printr-o valoare > 0.

- **Pixelii de umbra sunt afisati in culorile fragmentelor vizibile, calculate cu formula.**

Cfragment = Cemisiva + Cambianta = ke + ka*AmbientLight

ke, ka – proprietati de material

Cemisiva: o culoare de umbra

– Se presupune ca s-au calculat poligoanele de umbra.

– Conturul fiecarui poligon de umbra trebuie sa fie orientat in sens trigonometric atunci cand este privit din exteriorul volumului de umbra. In acest fel, poligoanele PUV sunt cele “din faţa” observatorului, iar poligoanele PUN sunt cele “din spate”, atunci cand volumul de umbra este privit din exterior.

VOLUME DE UMBRA (8)

Implementarea metodei folosind stencil-buffer(3)

Consideram cazul in care observatorul nu este in umbra (cazul uzual).

Pasul 1

Initializeaza z-buffer si stencil-buffer; // nu trebuie programate, acestea sunt
// initializate in mod automat pentru fiecare cadru imagine

Activeaza scrierea in z-buffer si dezactiveaza scrierea in stencil-buffer ; //implicite

Se trimite scena in banda grafica: **scena este rasterizata fara lumini (folosindu-se algoritmul z-buffer)**

Efectul:

– in frame-buffer se obtine imaginea scenei in umbra (fara lumini). Culoarea fiecarui pixel este:

Cpixel = culoarea emisiva + culoarea ambianta;

La sfarsitul acestui pas, z-buffer contine coordonatele z ale fragmentelor vizibile in imagine.

VOLUME DE UMBRA (9)

Pasul 2 **Implementarea metodei folosind stencil-buffer(4)**

glDepthMask(GL_FALSE); // Dezactiveaza scrierea in z-buffer

glColorMask(GL_FALSE, GL_FALSE, GL_FALSE, GL_FALSE); // Dezactiveaza scrierea in frame-buffer

glFrontFace(GL_CCW);// feţele “din faţă” sunt orientate trigonometric

glCullFace(GL_BACK);//se vor elimina feţele din spate (nu sunt rasterizate poligoanele PUN)

glStencilFunc(GL_ALWAYS, 0x0, 0xff);// se activeaza scrierea in stencil buffer

glStencilOp(GL_KEEP, GL_KEEP, GL_INCR); //se incrementeaza stencil buffer pt toate
// fragmentele care trec testul de adancime

(https://learnopengl.com/Advanced-OpenGL/Stencil-testing)

Trimite in banda grafica poligoanele de umbra

Efectul:

Sunt rasterizate poligoanele de umbra de tip PUV.

Pentru fiecare fragment rezultat, f, se face testul:

daca f.z < z-buffer[f.y][f.x] // fragmentul f este in fața fragmentului vizibil in pixelul (x,y)

atunci stencil-buffer[f.y][f.x]++; // se intra intr-un volum de umbra:
// se incrementeaza numarul poligoanelor PUV
// aflate in fața fragmentului vizibil in (x,y)

VOLUME DE UMBRA (10)

Implementarea metodei folosind stencil-buffer(5)

Pasul 3

glCullFace(GL_FRONT);//se vor elimina feţele din faţă (nu sunt rasterizate poligoanele PUV)

glStencilOp(GL_KEEP, GL_KEEP, GL_DECR); //se decrementeaza stencil buffer pt toate // fragmentele care trec testul de adancime

Trimite in banda grafica poligoanele de umbra

Efectul:

Sunt rasterizate poligoanele de umbra de tip PUN.

Pentru fiecare fragment rezultat, f, se face testul:

daca f.z < z-buffer[f.y, f.x] // f este in faţa fragmentului vizibil in pixelul (x,y)

atunci stencil-buffer[f.y, f.x]--; // se iese din volumul de umbra:
// se decrementeaza nr. de poligoane PUV aflate in fața fragm. vizibil in (x,y)

In stencil-buffer s-a obtinut o masca cu “gauri” (stencil-buffer[f.y, f.x] =0) in punctele care primesc lumina.

stencil-buffer[f.y, f.x] > 0 daca

nr. de poligoane PUV aflate in fața fragmentului vizibil in (x,y) > nr. poligoane PUN

VOLUME DE UMBRA (11)

Implementarea metodei folosind stencil-buffer(6)

Pasul 4

```
glEnable(GL_STENCIL_TEST);// Activeaza testul stencil
glStencilOp(GL_KEEP, GL_KEEP, KEEP); // Dezactiveaza scrierea in stencil-buffer
glStencilFunc(GL_EQUAL, 0x0, 0xff); // testul stencil: pixelul(x,y) se modifica daca
                                    // stencil-buffer [y][x]=0
glDepthMask(GL_TRUE); // Activeaza scrierea in z-buffer
glColorMask(GL_TRUE, GL_TRUE, GL_TRUE, GL_TRUE); // Activeaza scrierea in frame-buffer
```

Trimite scena in banda grafica si calculeaza culorile fragmentelor (in Vertex shader/Fragment shader) tinand cont de sursele de lumina

Efectul:

Se rasterizeaza scena modificandu-se in frame-buffer numai culorile pixelilor (x,y) pentru care stencil-buffer [y][x] =0 (fragmentul nu este in umbra)

VOLUME DE UMBRA (12)

Implementarea metodei folosind stencil-buffer(7)

Cazul general:

Se initializeaza stencil-buffer cu o valoare reprezentand numarul de volume de umbra care contin pozitia observatorului.

Calculul numarului de volume de umbre care contin pozitia obsrvatorului poate fi complex

→In varianta “Depth fail”, numita si *Carmack’s reverse* (descoperita de John Carmack in dezvoltarea jocului Doom 3), nu conteaza pozitia observatorului.

- Varianta originala, prezentata inainte, mai este numita “Depth pass”, deoarece bufferul stencil este incrementat/decrementat atunci cand “testul de adancine trece”.

Carmack’s reverse (“Depth fail”)

Difera de varianta “Depth pass” prin felul in care este construita masca pixelilor care primesc lumina de la sursa: pasii 2 si 3.

VOLUME DE UMBRA (13)

Implementarea (“Depth fail”) (1)

Pasul 2

glDepthMask(GL_FALSE); // Dezactiveaza scrierea in z-buffer

glColorMask(GL_FALSE, GL_FALSE, GL_FALSE, GL_FALSE); // Dezactiveaza scrierea in frame-buffer

glFrontFace(GL_CCW);// feţele “din faţă” sunt orientate trigonometric

glCullFace(GL_FRONT);//se vor elimina feţele din față (nu sunt rasterizate polig PUV)

glStencilFunc(GL_ALWAYS, 0x0, 0xff);// se activeaza scrierea in stencil buffer

glStencilOp(GL_KEEP, GL_INCR, GL_KEEP); //se incrementeaza stencil buffer pt toate
// fragmentele care nu trec testul de adancime

Trimite in banda grafica poligoanele de umbra

Efectul:

Sunt rasterizate poligoanele de umbra de tip PUN.

Pentru fiecare fragment rezultat, f, se face testul:

daca !(f.z < z-buffer[f.y][f.x]) // fragmentul f este in spatele fragm vizibil in pixelul (x,y)

atunci stencil-buffer[f.y][f.x]++; // se incrementeaza nr. de poligoane PUN
// aflate in spatele fragmentului vizibil in (x,y)

VOLUME DE UMBRA (14)

Implementarea (“Depth fail”) (2)

Pasul 3

glCullFace(GL_BACK);//se vor elimina feţele din faţă (nu sunt rasterizate poligoanele PUN)

glStencilOp(GL_KEEP, GL_DECR, GL_KEEP); //se decrementeaza stencil buffer pt toate
// fragmentele care nu trec testul de adancime

Trimite in banda grafica poligoanele de umbra

Efectul:

Sunt rasterizate poligoanele de umbra de tip PUV.

Pentru fiecare fragment rezultat, f, se face testul:

daca !(f.z < z-buffer[f.y, f.x]) // f este in spatele fragmentului vizibil in pixelul (x,y)
atunci stencil-buffer[f.y, f.x]--; //se decrementeaza nr. de poligoane PUN
// aflate in spatele fragmentului vizibil in (x,y)

In stencil-buffer s-a obtinut o masca cu “gauri” (stencil-buffer[f.y, f.x]=0) in punctele care primesc lumina.

VOLUME DE UMBRA (15)

Aprecieri asupra metodei volumelor de umbra

- **Pozitive:**

➢ Este generală şi poate fi aplicată pentru orice scena, pentru oricate surse de lumina.

➢ Produce forme de umbre exacte.

- **Negative:**

➢ Complexa din punct de vedere al timpului de calcul:

se rasterizeaza scena de 2 ori si poligoanele de umbra o singura data dar in 2 pasi!

➢ Sursele de lumina sunt considerate punctiforme, de aceea nu pot fi obtinute penumbre.