Iluminarea globala
-Ray Tracing-

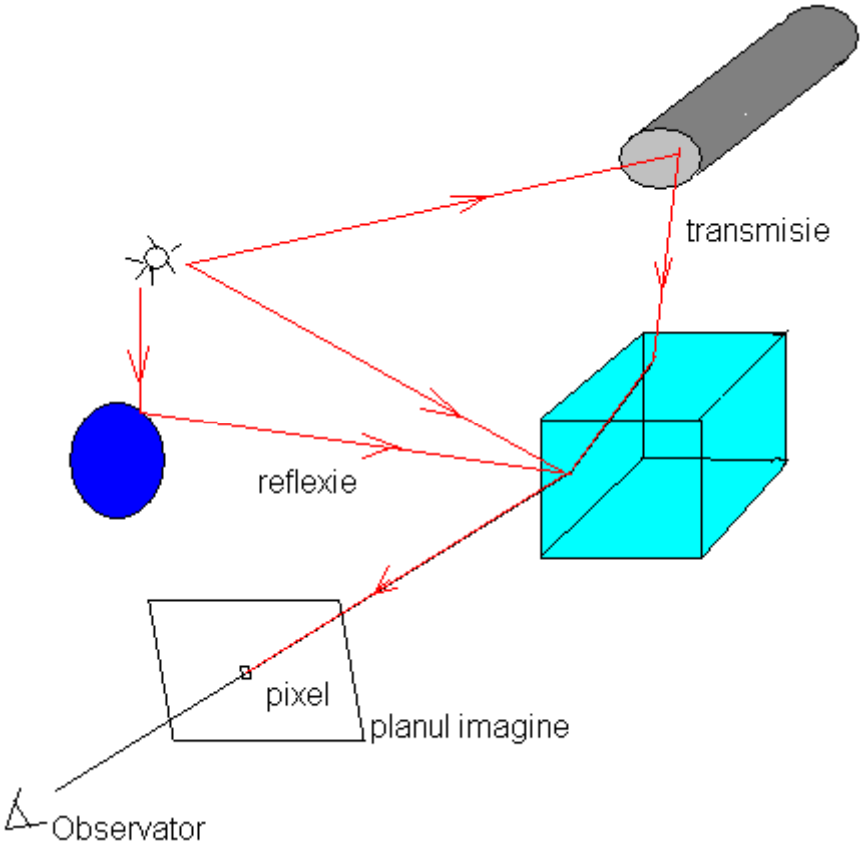
Prof. univ. dr. ing. Florica Moldoveanu

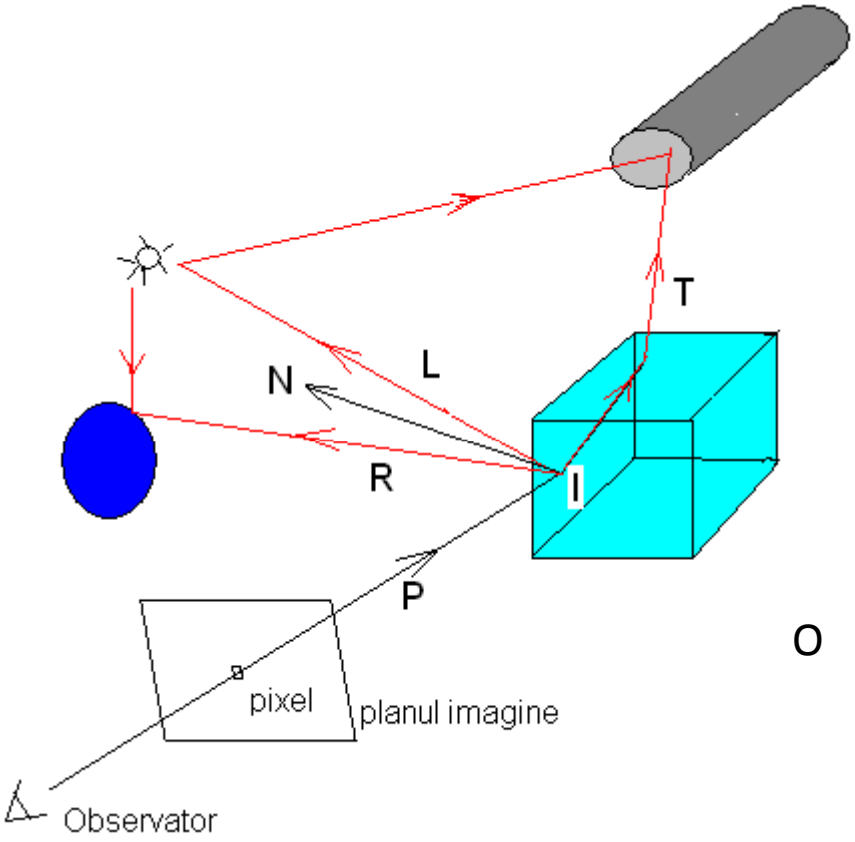
Iluminarea globala in Ray tracing(1)

Imbina:

- Eliminarea partilor nevizibile
- Calculul reflexiei luminii
- Calculul refractiei luminii
- Calculul umbrelor
- Interactiunea globala a reflexiei si refractiei luminii la nivelul intregii scene 3D

Iluminarea globala in Ray tracing(2)





O

Raze de lumina: directa si indirecte, de la o sursa de lumina pe o suprafata

In calculul culorii pixelului se considera raze cu directia inversa celor din realitate

Algoritmul Ray tracing

Imaginea se calculeaza pixel cu pixel, pornind de la spatiul imagine.

Considerand o singura sursa de lumina:

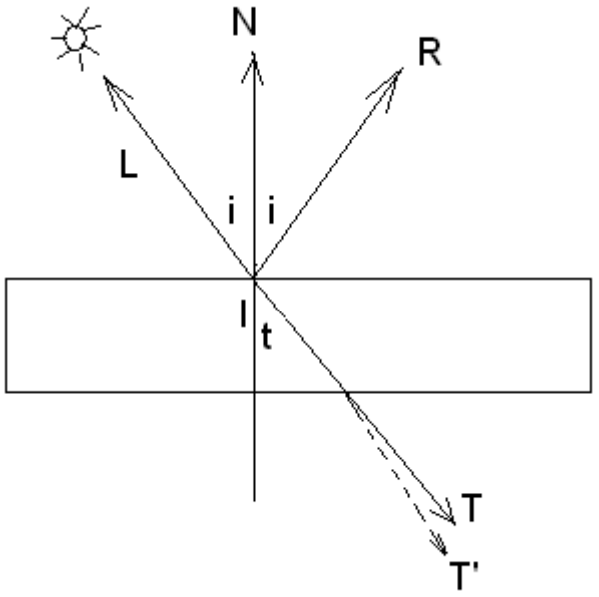
Pentru fiecare pixel al imaginii:

- Se calculeaza raza primara, P, care porneste din pozitia observatorului si trece prin centrul pixelului, in planul imaginii
- Se determina punctul de intersectie, I, al razei primare cu cel mai apropiat obiect de observator
- **Daca** raza nu intersecteaza nici un obiect al scenei,
  - Se afiseaza pixelul in culoarea fondului

  altfel
  - Se calculeaza culoarea obiectului in punctul I, tinand cont de:
    - Reflexia luminii provenita direct de la sursa de lumina, raza L
    - Lumina provenita in I prin reflexie speculara de la alte obiecte ale scenei, raza R
    - Lumina provenita in I prin transmisie de la alte obiecte ale scenei, raza T
  - Afiseaza pixelul in culoarea obtinuta prin combinarea contributiei celor trei raze

Ray tracing – calculul razelor

Calculul razelor care contribuie la reflexia luminii in punctul de intersectie cu raza primara



L: raza din I catre sursa de lumina

R: raza reflectata specular, simetrica, fata de N, cu L

$$R = 2(N \cdot L) \cdot N - L$$

T: raza transmisa, calculata pe baza legii lui Snell:

$n1/n2 = \sin(t)/\sin(i)$, n1,n2: indicii de refractie

$$T = L*(n1/n2) – (\cos(t) + (n1/n2)*(L \cdot N))*N$$

✓*Razele sunt considerate infinit subtiri*

✓*Reflexia speculara si refractia au loc fara imprastiere (sunt perfect focalizate)*

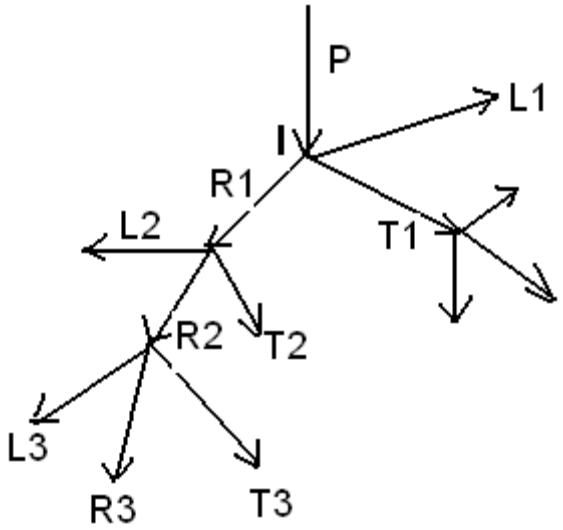
➢**Efect**: obiectele din imaginea produsa sunt de regula stralucitoare, producand reflexii multiple focalizate.

Ray tracing – arborele de raze

Algoritmul Ray tracing este recursiv:

Lumina provenita in punctul I prin reflexie speculara de la un obiect O1 sau prin transmisie de la un obiect O2, poate fi compusa din:

- Reflexia luminii provenita direct de la o sursa, de catre O1/O2
- Reflexia speculara a altor obiecte de catre O1/O2
- Transmisia luminii prin O1/O2 provenita de la alte obiecte ale scenei



Arborele de raze

➢*Pentru obtinerea culorii pixelului, se evalueaza arborele de raze de la frunze catre radacina*

Ray tracing- calculul reflexiei luminii(1)

Aproximarea reflexiei luminii in punctul I:

$I_\lambda(I) = I_{local\lambda}(I) + Ks*R_\lambda(I) + Kt*T_\lambda(I)$

unde:

λ – reprezinta lungimea de unda: expresia se evalueaza pentru R,G,B

$I_{local\lambda}(I)$ – reprezinta componenta rezultata prin reflexia luminii provenite direct de la sursele de lumina din scena 3D (calculata folosind modelul de iluminare locala)

Ks – este coeficientul de reflexie speculara al materialului obiectului

$R_\lambda(I)$ – reprezinta lumina provenita prin reflexie speculara in punctul I:

se obtine prin evaluarea arborelui de raze

Kt – este coeficientul de transmisie, specific materialului obiectului

$T_\lambda(I)$ - reprezinta lumina provenita prin transmisie (refractie)in punctul I:

se obtine prin evaluarea arborelui de raze

0<= ks, kt <= 1

Ray tracing- calculul reflexiei luminii(2)

Modelul de iluminare locala:

$I_{local\lambda}(I) = Ia_{\lambda} * Ka + Isursa_{\lambda} * fat * s * [kd * (N \cdot L) + lum * Ks * (N \cdot H)^{n}]$

Iaλ – reprezinta intensitatea luminii ambiante

Ka – este coeficientul de difuzie a luminii ambiante, specific materialului obiectului, 0<= ka <= 1

$Isursa_{\lambda}$ – reprezinta intensitatea luminii provenite de la sursa

L – este versorul directiei din punctul I catre pozitia sursei de lumina

N – este normala in punctul I (versor)

H – este versorul directiei bisectoare a unghiului dintre L si vectorul din I catre observator (raza P)

fat – este factorul de atenuare a luminii de la sursa, proportional cu distanta de la sursa la punctul I

0<= s <= 1,

s = 0, daca lumina de la sursa nu ajunge in punctul I: vectorul L intersecteaza un obiect opac al scenei

s = 1, daca vectorul L nu intersecteaza un alt obiect al scenei

0 < s <1, daca vectorul L intersecteaza un obiect transparent (sau mai multe)

Punctul I primeste lumina de la sursa daca:

- produsul scalar (N·L)>0 (lum =1, altfel =0)
- raza L nu intersecteaza un obiect opac al scenei

➢ Daca I nu primeste lumina de la sursa, se afiseaza in culoarea luminii ambiante.

Ray tracing- calculul reflexiei luminii(3)

- Daca in scena 3D exista mai multe surse de lumina, fiecare poate contribui in mod diferit la $I_{local\lambda}(I)$:

$$I_{local\lambda}(I) = Ia_{\lambda} * Ka + \sum_{i=1,n} Isursa_{i,\lambda} * fat_i * s_i * [kd * (N \cdot L_i) + lum_i * Ks * (N \cdot H_i)^n]$$

- Pentru evaluarea componentelor $R_{\lambda}(I)$ si $T_{\lambda}(I)$ din calculul culorii in I,

$$I_{\lambda}(I) = I_{local\lambda}(I) + Ks * R_{\lambda}(I) + Kt * T_{\lambda}(I)$$

se coboara in arborele de raze pana la un numar pre-specificat de nivele (energia luminoasa scade destul de repede!).

Algoritmul Ray tracing recursiv(1)

Algoritmul Ray tracing recursiv

Pentru fiecare pixel al imaginii

```
{   *calculeaza raza primara, P;
    *culoare_pixel = TraseuRaza(P, 1);
    *afiseaza pixelul in culoare_pixel;
}
```

Culoare TraseuRaza(Raza R, int n)

```
{   // n este nivelul in arborele de raze
    *calculeaza intersectiile razei R cu obiectele scenei;
    daca (nu exista intersectii), atunci
            return  (culoare_fond);
    altfel
            *fie I punctul de intersectie cel mai apropiat de observator si O obiectul intersectat;
            *calculeaza normala N, in punctul I;
            return  CuloarePunct(O, R, I, N, n);
}
```

Algoritmul Ray tracing recursiv(2)

```
Culoare CuloarePunct(Obiect O, Raza R, Punct I, Normala N, adancime_arbore n)
{   Culoare  culoare;
    culoare = culoare_ambianta;
    pentru fiecare sursa de lumina S executa
        *calculeaza vectorul L, din I catre S
        daca ((Nu·Lu) > 0 si vectorul L nu interscteaza un obiect opac al scenei) atunci
            *calculeaza contributia sursei S la culoare, CS
            culoare = culoare + CS
            daca (n< nivel_max) atunci
                daca (obiectul O produce reflexii speculare) atunci
                    *calculeaza raza reflectata in punctul I, RS;
                    culoare = culoare + Ks* TraseuRaza(RS, n+1);
                daca (obiectul O este transparent) atunci
                    *calculeaza raza transmisa (refractata) in punctul I, RT;
                    culoare = culoare + Kt* TraseuRaza(RT, n+1);
    return (culoare)
}
```

Algoritmul Ray tracing recursiv(3)

Deficientele algoritmului

1. **Efecte de aliasing**: razele sunt convergente si infinit subtiri
   - Defecte ale marginilor suprafetelor(marginile nu sunt drepte, au aspect zimțat)
   - Obiecte mici si subtiri pot sa apara si sa dispara din imagine, la schimbarea pozitiei observatorului

   Imbunatatire:

   – marirea rezolutiei esantionarii spatiului 3D; ex. 4 raze primare/pixel → creste complexitatea algoritmului

2. **Complexitatea computationala**
   - buna din punct de vedere teoretic:

     - pentru o raza/pixel este de O(p*n), unde p este nr. de pixeli - constant (independent de scena)

       = O(n), unde n este nr. de obiecte din scena
   - Creste liniar cu numarul de raze/pixel
   - Complexitatea calculelor de intersectie depinde de geometria obiectelor intersectate

   Imbunatatire:

   – Reducerea calculelor de intersectie, prin folosirea de volume incadratoare si gruparea obiectelor din scena 3D

   – Complexitatea se poate reduce la O(log n)

Algoritmul Ray tracing – optimizari(1)

Optimizari:

1. Imbunatatirea calitatii imaginii(1)

Marirea numarului de raze primare

1) Uniforma:

4raze/pixel, care trec prin colturile suprafetei pixelului

- culoarea pixelului se determina ca medie a culorilor obtinute cu cele 4 raze primare
- razele primare utilizate pentru un pixel contribuie la culoarea pixelilor adiacenti
- pentru o imagini de mxn pixeli: (m+1)x(n+1) raze→ numarul de raze creste cu (m+n+1)

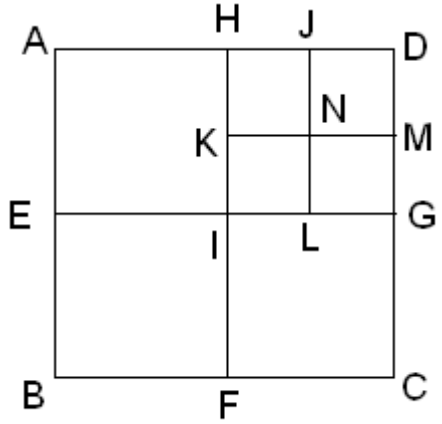
2) **Adaptiva** (Whitted):

- Se creste rezolutia esantionarii spatiale numai in zonele in care este necesara operatia de anti-aliasing:
- Daca diferenta dintre culorile celor 4 raze primare pentru un pixel este mare, se subdivizeaza suprafata pixelului in 4 subzone
- Se aplica acelasi criteriu de comparatie intre culorile celor 4 raze primare corespunzatoare unei subzone
- Subdivizarea se continua recursiv pana la un nivel maxim prestabilit sau pana cand diferenta dintre cele 4 culori scade sub un prag dat
- Culoarea pixelului se obtine ca o medie ponderata a culorilor subzonelor in care a fost divizata suprafata pixelului

Algoritmul Ray tracing – optimizari(2)

Imbunatatirea calitatii imaginii(2)

Exemplu de subdivizare adaptiva



CA, CB, CC, CD, CE, CF, CG, CH, CI, CJ, CK, CL, CM, CN

- sunt culorile obtinute cu razele primare care trec prin punctele respective

Culoare pixel (A-B-C-D) =

¼((CA+CE+CI+CH)/4 + (CE+CB+CF+CI)/4 + (CI+CF+CC+CG)/4 +

¼((CH+CK+CN+CJ)/4 + (CK+CI+CL+CN)/4 + (CJ+CN+CM+CD)/4 +

(CN+CL+CG+CM)/4))

Algoritmul Ray tracing – optimizari(3)

Optimizari:

2. Reducerea complexitatii computationale(1)

2.1 Utilizarea de volume incadratoare la nivel de obiect/ grup de obiecte

- Testarea intersectiei raza-volum incadrator in loc de raza-obiect:
  - Daca raza nu intersecteaza volumul incadrator → nu se va calcula intersectia cu obiectul/obiectele din volum
- Eliminarea unui intreg grup de obiecte care nu este intersectat de raza
- Evitarea calculelor de intersectie cu obiectele scenei, care pot avea geometrie complexa

Volume incadratoare: sfera, paralelipipedul cu fetele paralele cu planele principale, elipsoidul, cilindrul.

- Calculul intersectiei raza-volum incadrator trebuie sa fie mai simplu decat calculul intersectiei cu obiectul (de ex. o retea poligonala)
- Volumul incadrator al unui obiect se alege in functie de forma obiectului

Algoritmul Ray tracing – optimizari(4)

Reducerea complexitati computationale(2)

Calcule de intersectie raza-volum incadrator(1)

Ecuatia razei: r(t) = P0 + t*D ,

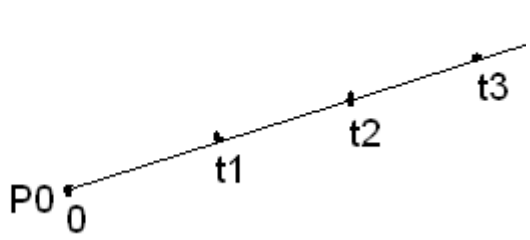
P0 – este originea razei

D – este directia razei

$x = x0 + t*dx$

$y = y0 + t*dy$

$z = zo + t*dz$



Punctul cel mai apropiat de originea razei este r(tmin)

1) Intersectia cu sfera

$(x-a)^2 + (y-b)^2 + (z-c)^2 = r^2$

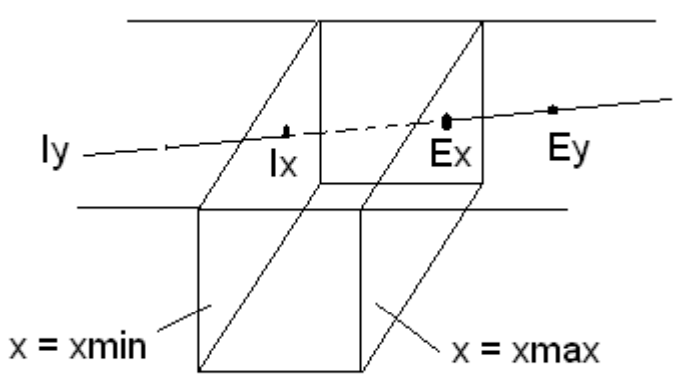
- Se inlocuiesc in ec sferei x, y, z, cu cele din ecuatia razei
- Rezulta o ecuatie de grad 2 in t
- Se calculeaza discriminantul D, al ecuatiei:
  - D<0 – raza nu intersecteaza sfera
  - Daca sfera este obiect al scenei (intereseaza punctul de intersectie):
    - D=0 – raza este tangenta la sfera
    - D>0 – se calculeaza radacinile, t1, t2
      - punctul de intersectie mai apropiat de observator este r(tmin(t1,t2))

Algoritmul Ray tracing – optimizari(5)

Reducerea complexitati computationale(3)

Calcule de intersectie raza-volum incadrator(2)

2) Intersectia cu un paralelipiped cu fetele paralele cu planele principale



Planele care delimiteaza volumul:

x= xmin, x = xmax, y=ymin, y = ymax, z= zmin, z= zmax

Intersectia cu x=xmin si x=xmax

xmin = x0 + t*dx → t1x = (xmin – x0)/dx

xmax = x0 + t*dx →t2x =(xmax – x0)/dx

tix = min(t1x, t2x), tex = max(t1x, t2x)

Ix = r(tix), Ex = r(tex)

Analog pentru intersectia cu planele y=ymin si y=ymax:

tiy = min(t1y, t2y), tey = max(t1y, t2y)

Iy = r(tiy), Ey = r(tey)

Raza intersecteaza volumul daca:

max(tix, tiy) < min (tex, tey)

Algoritmul Ray tracing – optimizari(6)

Reducerea complexitati computationale(4)

Calcule de intersectie raza-volum incadrator(3)

2) Intersectia cu un paralelipiped cu fetele paralele cu planele principale (continuare)

daca max(tix,tiy) > min(tex,tey) → raza nu intersecteaza volumul

altfel, se calculeaza intersectia cu z=zmin si z= zmax

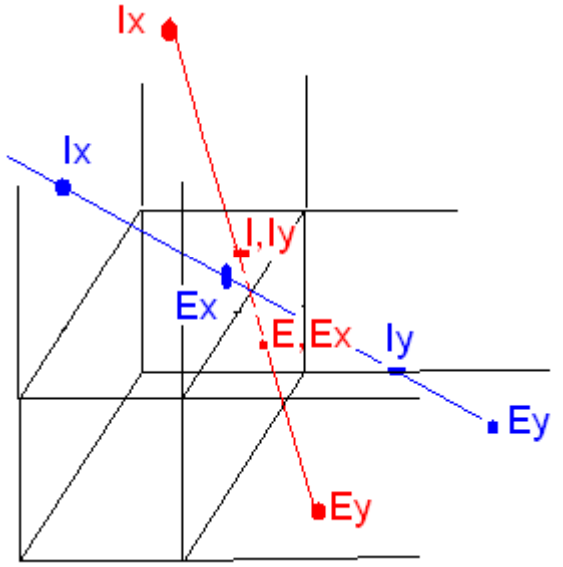
Punctele de intrare si iesire din volum sunt:

I = r(max(tix, tiy, tiz))

E = r(min(tex,tey,tez)

daca max(tix, tiy, tiz) < min(tex, tey, tez)

atunci raza intersecteaza volumul



max(tix, tiy) = tiy , min(tex,tey)=tex

max(tix,tiy) > min(tex,tey)

max(tix,tiy) < min(tex,tey)

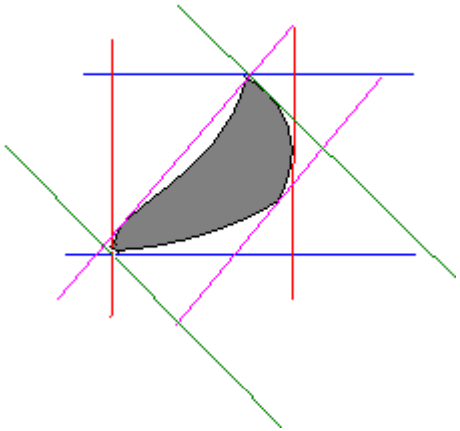
Algoritmul Ray tracing – optimizari(7)

Reducerea complexitatii computationale(5)

Calcule de intersectie raza-volum incadrator(4)

3) Intersectia cu un volum incadrator poliedru convex

Volumul incadrator (Kay, Kajiya[1986]) este un poliedru convex format din intersectiile a 4 perechi de plane paralele, inclinate la 0, 45, 90, 135 grade fata de planul orizontal.



Ec. unui plan: A*x + B*y + C*z + D =0

- Fie t11 si t12 - valorile pentru intersectiile cu 2 perechi de plane paralele

  t1min = min(t11,t12) corespunde punctului de intersectie mai apropiat de observator
- Fie t21, t22 - valorile pentru intersectiile cu urmatoarele 2 perechi de plane paralele
- daca max (t1min, t2min) > min (t1max, t2max) raza nu intersecteaza volumul

  altfel, se continua cu intersectia urmatoarei perechi de plane

➢Punctele de intersectie cu volumul: I = r(max($t_i$min)), E = r(min($t_i$max))

Algoritmul Ray tracing – optimizari(8)

Reducerea complexitatii computationale(6)

2.2. Divizarea scenei in volume incadratoare(1)

Divizarea regulata a scenei

- se porneste de la paralelipipedul incadrator al scenei, care se divizeaza recursiv in (8) subvolume egale, numite voxeli, pana la o anumita rezolutie
- la divizare nu se tine cont de structura scenei
- pentru fiecare voxel se memoreaza lista obiectelor pe care le contine
- daca raza nu intersecteaza un voxel → ea nu intersecteaza nici un obiect atasat voxelului
- volumul de voxeli poate fi reprezentat eficient, tinand cont de coerenta spatiala a voxelilor
- traseul razei prin volumul de voxeli poate fi calculat eficient printr-un algoritm DDA 3D

Algoritmul Ray tracing – optimizari(9)

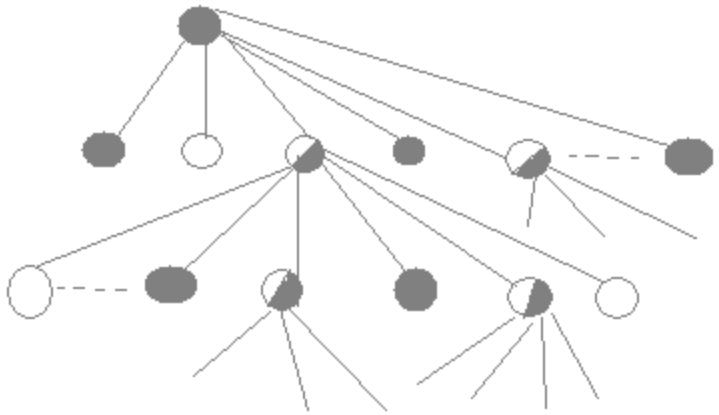
Reducerea complexitatii computationale(7)

2.2 Divizarea scenei in volume incadratoare (2)

Divizarea adaptiva a scenei

- Se porneste de la paralelipipedidul incadrator al scenei, care se divide in 8 subvolume egale, in mod recursiv, divizarea unui subvolum terminandu-se daca el nu contine nici un obiect al scenei sau contine un singur obiect.
- Scena se reprezinta printr-un arbore octal, fiecare nod fiind asociat unui subvolum.
- Pentru intersectia razei cu scena se fac teste de intersectie de la radacina spre frunze.
- Daca un subvolum nu este intersectat de raza, atunci nici unul dintre subvolumele (obiectele) din subarborele nodului subvolumului nu va fi intersectat.

Arborele octal al scenei



Algoritmul Ray tracing – optimizari(10)

Reducerea complexitatii computationale(8)

2.3 Ierarhie de volume incadratoare de grupuri de obiecte

- Scena 3D este reprezentata printr-un arbore (arborele scenei) in care fiecare nod are un parinte si un numar oarecare de copii
- Frunzele contin obiectele scenei, celelalte noduri contin grupuri de obiecte
- Fiecarui nod ii este atasat volumul incadrator al grupului de obiecte
- Pentru intersectia razei cu scena este parcurs arborele de la radacina spre frunze