Decuparea vectorilor

Prof. univ. dr. ing. Florica Moldoveanu

Curs *Elemente de Grafică pe Calculator* – UPB, Automatică şi Calculatoare
2020-2021

Decuparea primitivelor grafice

Decuparea: necesara pentru obiectele partial vizibile in imagine

Intr-un sistem grafic 2D:

— Decupare la frontiera unui dreptunghi din spatiul de afisare, avand laturile paralele cu cele ale sistemului de coordonate carteziene 2D (viewport)

— Decupare la frontiera unui poligon oarecare din spatiul de afisare

— Decuparea se poate efectua:

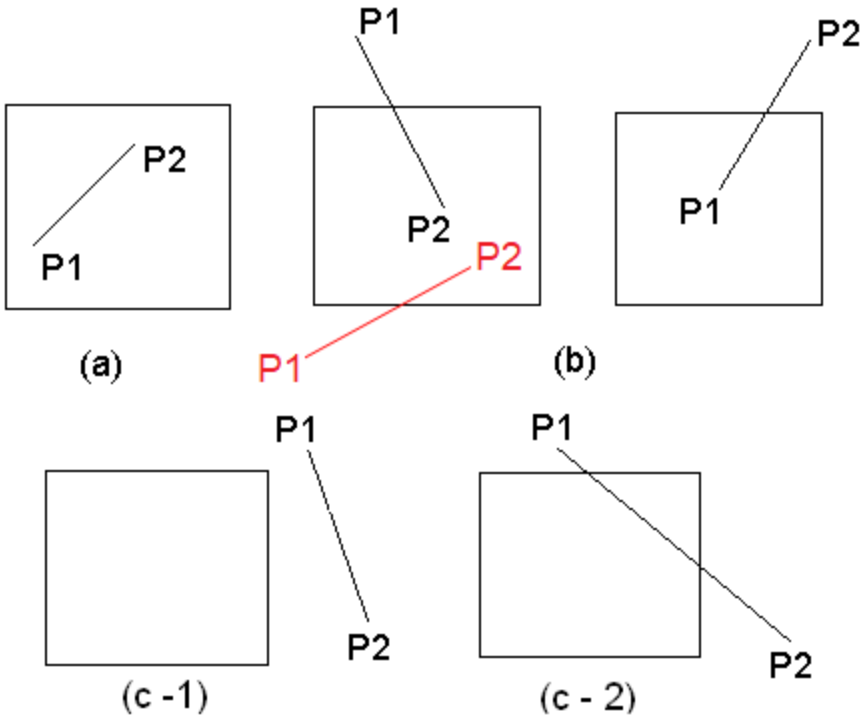
- Analitic, prin intersectia primitivelor grafice cu frontiera zonei de decupare (dreptunghi/poligon oarecare), inainte de rasterizare
- La momentul rasterizarii primitivelor, prin testul de apartenenta a fiecarui fragment la dreptunghiul/poligonul de decupare – de ex. pentru cercuri si elipse

— Pentru linii si poligoane, decuparea se efectueaza analitic (mai eficient).

Intr-un sistem grafic 3D bazat pe OpenGL:

— Decuparea se efectueaza la frontiera volumului vizual canonic (dupa transformarea de proiectie, inainte de transformarea in viewport).

Decuparea vectorilor 2D(1)



Pozitia vectorului fata de dreptunghiul de decupare

Presupunem urmatoarea ordine a calculelor de intersectie a unui vector P1-P2 cu dreptunghiul:

a. Calcul intersectie cu latura din stanga
b. Calcul intersectie cu latura de sus
c. Calcul intersectie cu latura din dreapta
d. Calcul intersectie cu latura de jos

In cazul vectorului rosu, calculele din pasii a, b, c sunt inutile.

Optimizari: **evitarea calculelor de intersectie inutile**

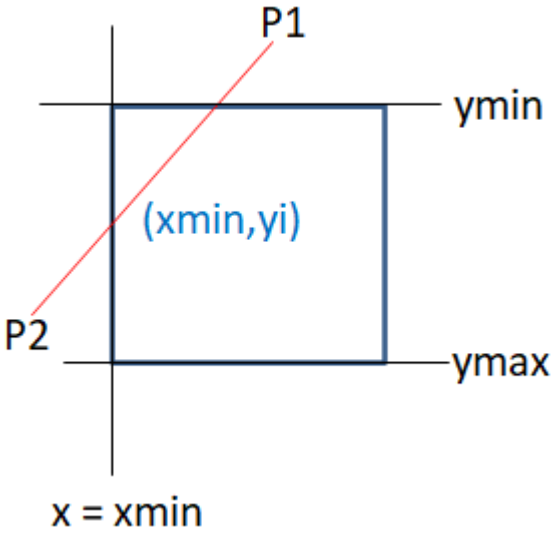
- Evitarea calculelor de intersectie in cazurile de tip (c-1):

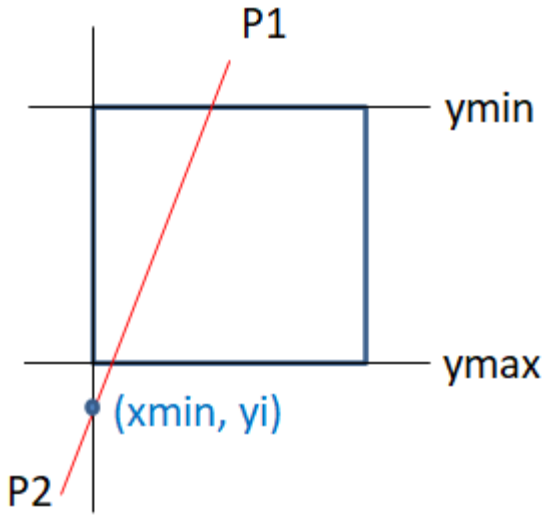
  - cum se diferentiaza cazurile (c-1) de cazurile (c-2) fara a efectua calcule de intersectie?

- Efectuarea calculelor de intersectie **numai** pentru laturile intersectate: cazurile (b) si (c-2).

Decuparea vectorilor 2D(2)

Intersectia segmentului P1-P2 cu latura x = xmin a dreptunghiului





Ec. parametrica a vectorului P1-P2:

P(t) = P1 + (P2-P1)*t 0<= t <=1 intre P1-P2

Punctul de intersectie cu dreapta x=xmin are abscisa xmin:

1) xmin=x1+t(x2-x1) → t=(xmin-x1)/(x2-x1)

2) este 0<=t<=1 ???

da: punctul de intersectie se afla pe segmentul P1-P2

- se calculeaza yi=y1+t*(y2-y1) = y1 + (xmin-x1)*(y2-y1)/(x2-x1)
- este ymin<= yi <= ymax ??

da: exista intersectie intre vectorul P1-P2 si latura x=xmin

nu: punctul de intersectie se afla pe prelungirea laturii

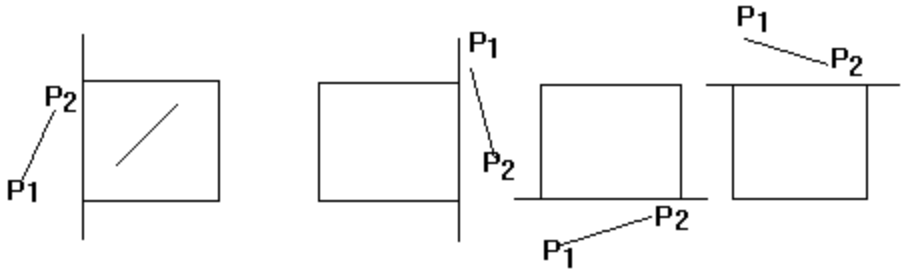
→tot calculul este inutil

Algoritmul Cohen-Sutherland (1)

1001	1000	1010
0001	0000	0010
0101	0100	0110

- Punctele din plan sunt codificate prin 4 biti, in functie de pozitia lor fata de dreptunghiul de decupare. De exemplu: b3b2b1b0
- b0=1, numai pentru punctele din stanga dreptunghiului,
- b1=1: dreapta, b2=1: sub dreptunghi, b3=1: deasupra dreptunghiului

Fie **cod1** codul binar al punctului P1 si **cod2** codul binar al punctului P2.



1) cod1 ==0 && cod2==0 :
segment acceptat
(P1 si P2 in interiorul dreptunghiului)

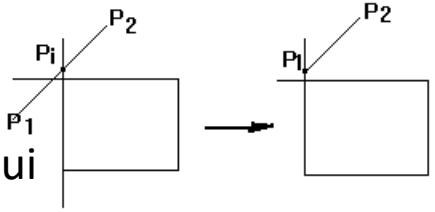
2) cod1 & cod2 != 0: segment rejectat
(cele 2 coduri au acelasi bit =1; P1 si P2 sunt de aceeasi parte a dreptunghiului)
segmentul nu intersecteaza dreptunghiul

3) altfel: se intersecteaza segmentul P1-P2 cu dreptunghiul de decupare

Algoritmul Cohen-Sutherland (2)

- Alegerea laturii de intersectie cu segmentul P1-P2: pe baza codului varfului P1, care trebuie sa fie situat in afara dreptunghiului de decupare.
- Daca cod1 = 0 → se interschimba P1 cu P2

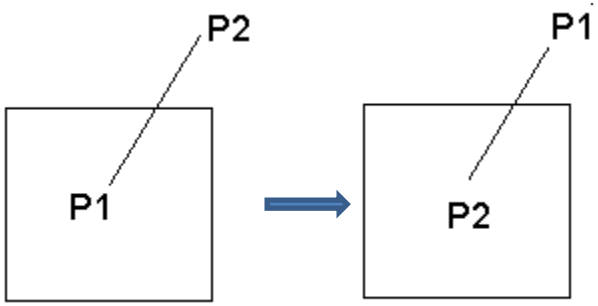
cod1 & 1 != 0
P1 in stanga dreptunghiului



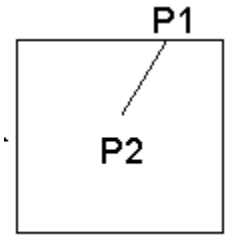
- Se muta P1 in punctul de intersectie
- In iteratia urmatoare cod1&cod2 != 0

→ segmentul este rejectat

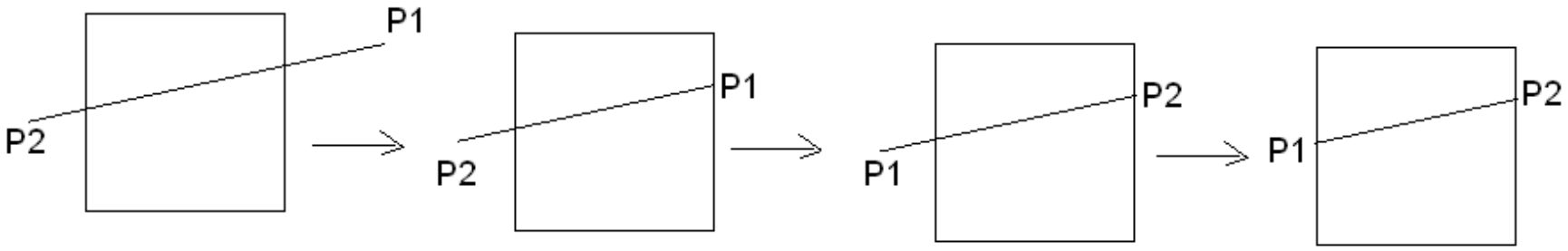
cod1 = 0



cod1 & 8 != 0
P1 deasupra dreptunghiului



Algoritmul Cohen-Sutherland (3)



Calculul punctelor de intersectie

$P(t) = P1 + (P2-P1)*t \rightarrow x(t) = x1 + t(x2-x1);\ y(t) = y1 + t(y2-y1)$

Intersectia cu dreapta y = ymax

$ymax=y1+t(y2-y1)$ de unde, $t=(ymax-y1)/(y2-y1)$

$x=x1+(ymax-y1)*(x2-x1)/(y2-y1) \rightarrow$ **$xi=x1+(ymax-y1)/m$** m: panta segmentului P1-P2

Intersectia cu dreapta x = xmin

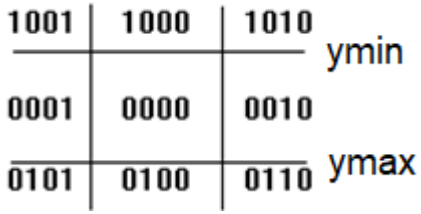
$xmin=x1+t(x2-x1)$, deci $t=(xmin-x1)/(x2-x1)$

$y=y1+(xmin-x1)*(y2-y1)/(x2-x1) \rightarrow$ **$yi=y1+(xmin-x1)*m$**

Algoritmul Cohen-Sutherland (4)

```
int xmin, ymin, xmax, ymax;  // colturile dreptunghiului de decupare

// functia calculeaza codul binar al unui punct din plan
int codif(int x, int y)
{
  int cod =0;
  if(x < xmin)  cod =1;
  if(x > xmax) cod=2;
  if(y > ymax)  cod |=4;  //b2=1
  if(y < ymin) cod |=8;  //b3=1
  return  cod;
}
```



Algoritmul Cohen-Sutherland (5)

```
int Cohen_Suth(int x1i, int y1i, int x2i, int y2i, int* x1d, int* y1d, int* x2d, int* y2d)
{ // primeste coord. capetelor vectorului de decupat si intoarce coord. partii din vector
  // inclusa in dreptunghi, daca exista
  int acceptat,rejectat,vertical;
  float m, x1=x1i, y1=y1i, x2=x2i, y2=y2i;
  int cod1, cod2, t;
  acceptat=rejectat=vertical=0;
  if (x1i != x2i)  //vectorul nu este vertical
      m=(y2-y1)/(x2-x1);
  else
      vertical=1;
 do
   { cod1=codif(x1,y1);  cod2=codif(x2,y2);
     if(cod1==0 && cod2==0) { acceptat=1; break;}
     if((cod1 & cod2) !=0 )  { rejectat=1; break;}
```

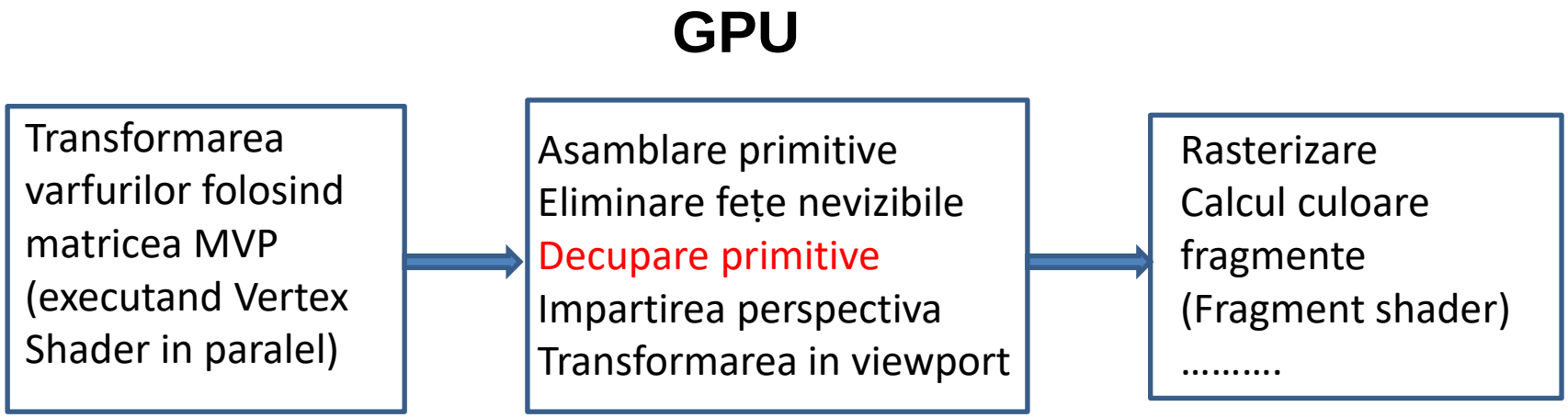
Algoritmul Cohen-Sutherland (6)

```
if(cod1==0) // P1 este in interiorul dreptunghiului de decupare
    {t=x1;x1=x2;x2=t; t=y1;y1=y2;y2=t; cod1=cod2;} //interschimbare capete
if(cod1 & 1)  //P1 in stanga dreptunghiului
     { y1+=(xmin-x1)*m;  x1=xmin;}  // punctul de intersectie devine P1 pentru iteratia urmatoare
  else
   if(cod1 & 2) //P1 in dreapta dreptunghiului
     { y1+=(xmax-x1)*m;  x1=xmax; }
      else
     if(cod1 & 4) // P1 sub dreptunghi
       { if (!vertical)   x1+=(ymax-y1)/m; //intersectia cu dreapta y = ymax
         y1=ymax;
       }
     else
       {if (!vertical) x1+=(ymin-y1)/m;
        y1=ymin;
       }
  }   while ( acceptat==rejectat); // cat timp vectorul nu este acceptat sau rejectat
```

Algoritmul Cohen-Sutherland (7)

```
 if(acceptat)
  {*x1d=x1;*y1d=y1;*x2d=x2;*y2d=y2;
   return 1;
  }
  return 0;
 }
}
```

Decuparea în banda grafica (1)



- Varfurile primitivelor pot avea asociate atribute: culoare, normala, coordonate textura
- Din decupare rezulta noi varfuri
- Pentru varfurile rezultate din decupare se calculeaza atribute prin interpolarea atributelor asociate varfurilor, folosind valoarea variabilei parametrice (t) în punctul de decupare.

Decuparea în banda grafica (2)

Transformarea de proiectie:

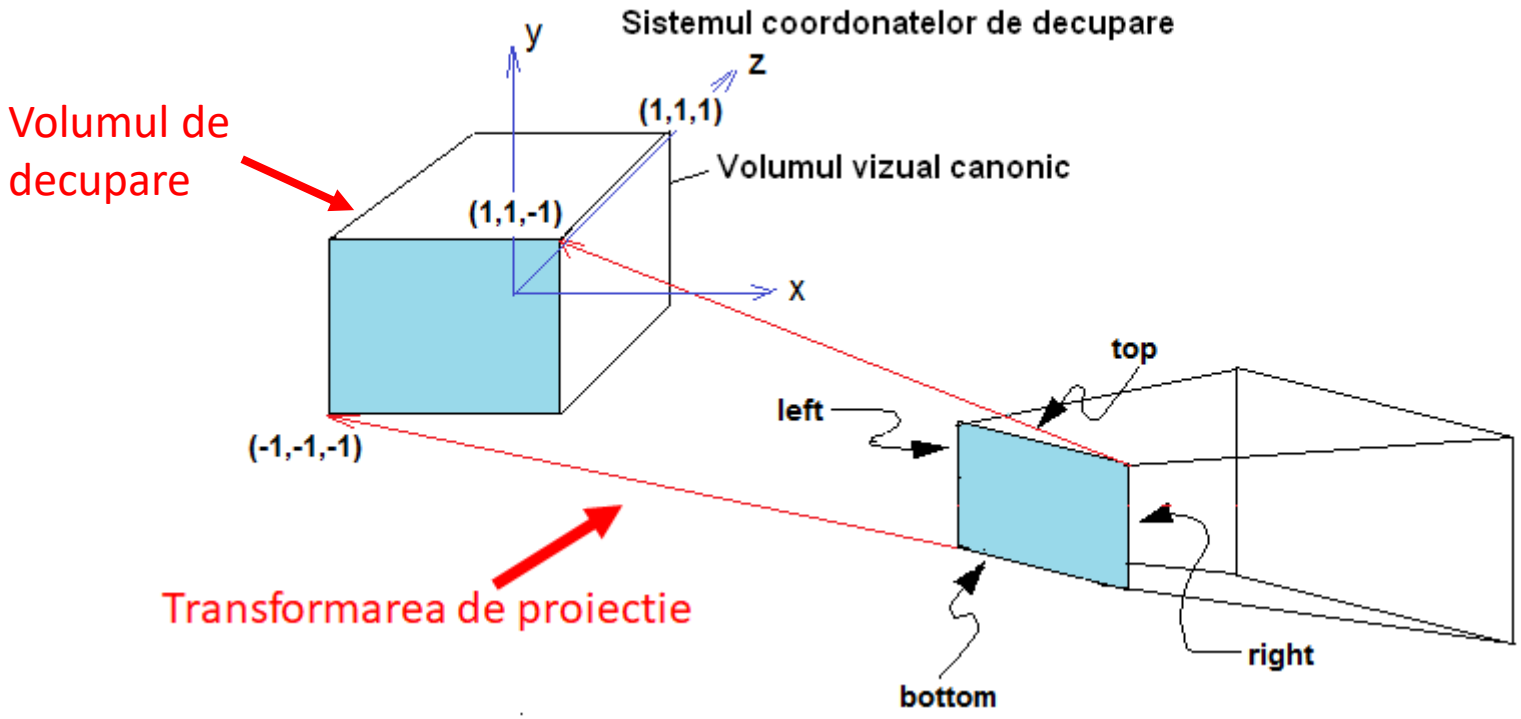
$[xc\ yc\ zc\ w]^T = P \bullet [xe\ ye\ ze\ 1]^T$

(clip coordinates) ← (eye coordinates)

Coordonate omogene → împartirea perspectiva → coordonate carteziene:

xd = xc/w , yd = yc/w , zd= zc/w , w = -ze

(xd, yd, zd): coordonate dispozitiv normalizate (**N**ormalized **D**evice **C**oordinates) raportate la sistemul coordonatelor de decupare



Decuparea în banda grafica (3)

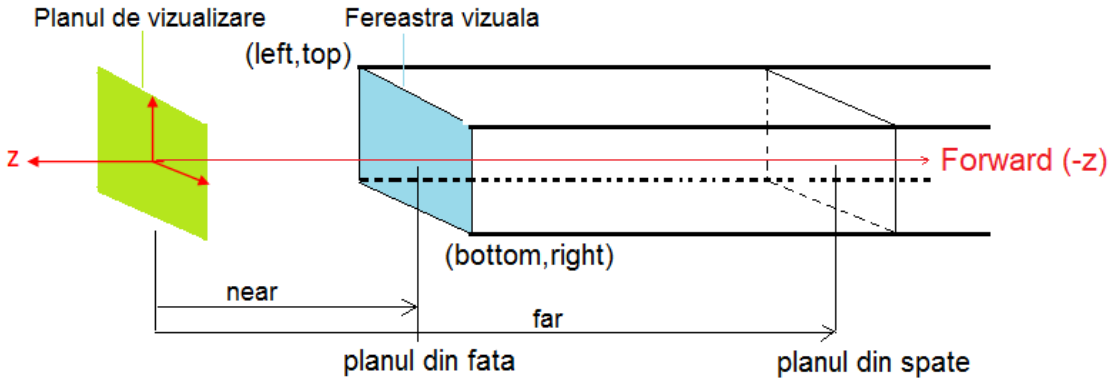
Decuparea: înainte sau dupa împarţirea perspectivă?

1. Inainte: decupare in coordonate omogene (mai complicata decat in coord. carteziene)
2. Dupa: decupare in coordonate carteziene (NDC) la marginile volumului vizual canonic

➢ Problema: cazul w=0 - impartirea prin zero

xd = xc/w , yd = yc/w , zd= zc/w , w = -ze

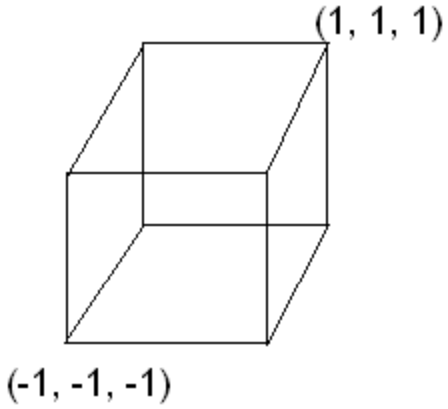
w=-ze =0: daca punctul transformat (xe ye ze) se afla in planul camerei



- Problema nu apare in cazul 1, caci varfurile din planul camerei sunt eliminate la decupare, ele nefiind incluse in volumul vizual: near>0, fața z=near → fața z=-1.
- In cazul 2, la conversia in NDC trebuie tratat cazul special w=0.

Cel mai simplu: w←0.000001 pt a evita impartirea prin zero, caci varful este oricum eliminat prin decupare, iar eroarea introdusa este neglijabila.

Generalizarea algoritmului Cohen-Sutherland pentru decuparea vectorilor 3D(1) (decuparea în coordonate carteziene)



Faţă de volumul vizual canonic, un punct din spatiu se poate afla: in interior, in stanga, in dreapta, sub volum, deasupra volumului, in fata, in spate:

→pentru codificarea pozitiei unui punct din spatiu fata de volumul vizual canonic sunt necesari 6 biti.

De ex. se poate face conventia:

b0 = 1 pentru puncte (x,y,z) cu x < -1

b1 = 1 pentru puncte (x,y,z) cu x > 1

b2 = 1 pentru puncte (x,y,z) cu y < -1

b3 = 1 pentru puncte (x,y,z) cu y > 1

b4 = 1 pentru puncte (x,y,z) cu z < -1

b5 = 1 pentru puncte (x,y,z) cu z > 1

daca: cod(P1) ==0 && cod(P2)==0
    segment acceptat

daca (cod(P1) & cod(P2)) != 0
    segment rejectat (P1, P2 de aceeasi parte a volumului vizual)
altfel
    se intersecteaza segmentul cu volumul

Generalizarea algoritmului Cohen-Sutherland pentru decuparea vectorilor 3D(2)

Intersectiile segmentului cu volumul vizual

Ec. parametrice ale segmentului

$x = x1 + t(x2 – x1)$

$y = y1 + t(y2 –y1)$

$z = z1 + t(z2 – z1)$

P1-P2 se intersecteaza cu o față a volumului, aleasa pe baza codului varfului P1

daca cod(P1) & 8 !=0 //P1 deasupra volumului

se efectueaza intersectia cu planul y= 1:

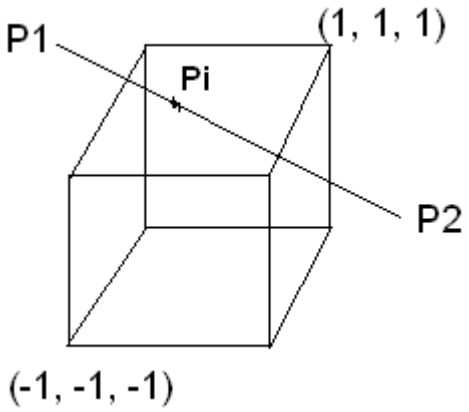
$1 = y1 + t(y2 – y1) \rightarrow tinters = (1-y1)/(y2-y1)$

//calculeaza coord punctului Pi si muta P1 in Pi

$x1 = x1 + tinters(x2 – x1)$

$y1 = 1;$

$z1 = z1 + tinters(z2 – z1)$



Generalizarea algoritmului Cohen-Sutherland pentru decuparea vectorilor 3D(3)