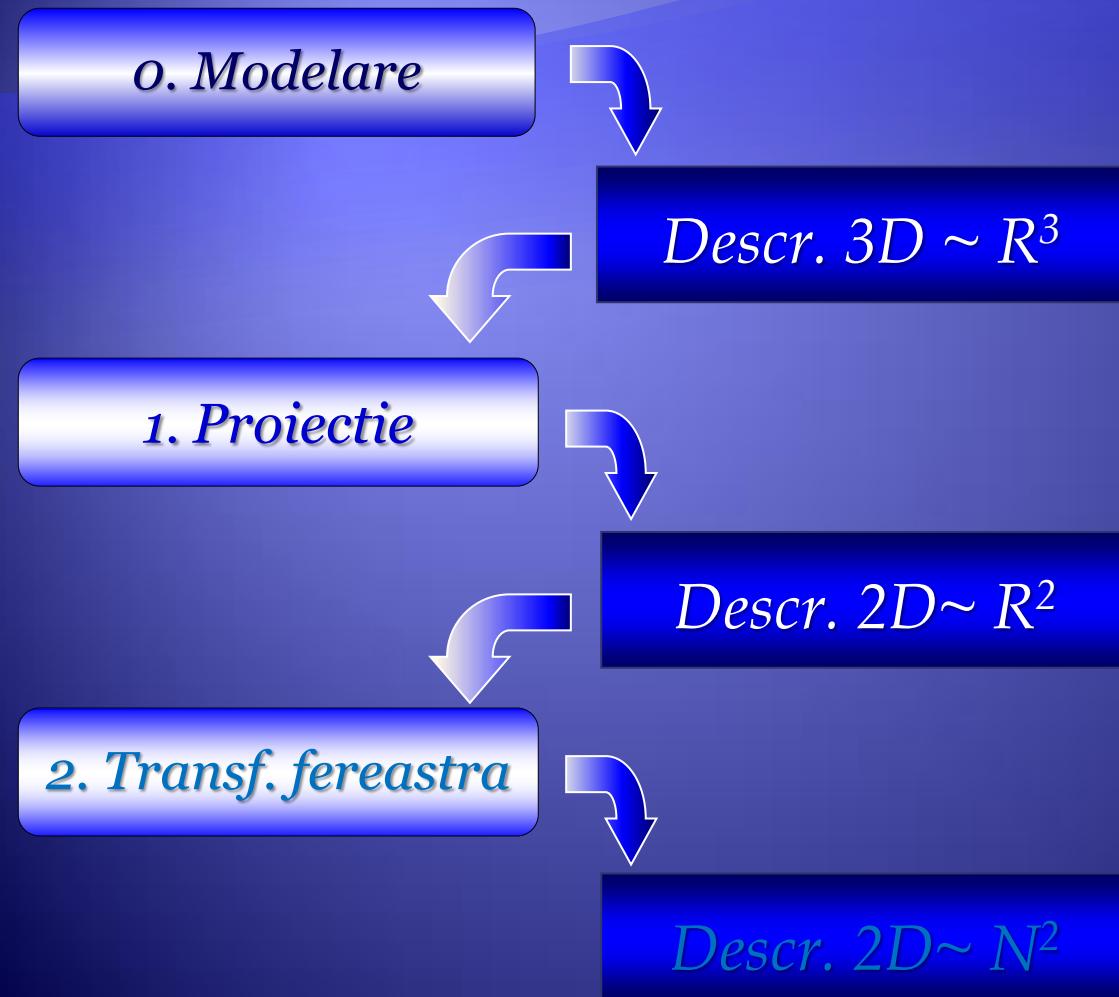


Grafică pe calculator (MLR5060)

Elemente de grafică 3_D

1. Transformări geometrice uzuale;
2. Reprezentarea curbelor, suprafețelor și corpurilor;
3. Observarea unui sistem 3_D de puncte;
4. Modelarea corpurilor;
5. Creșterea realismului imaginilor tridimensionale.

Etape in reprezentarea Ob. 3D



Spatiul 3D

*Fereastra Reală
Window*

*Fereastra Ecran
Viewport*

Transformări geometrice uzuale

Printr-o transformare a unui punct $P(x,y,z) \in R^3$ se va obține un alt punct $P'(x',y',z') \in R^3$.

a) Translație

Prin translația cu $(\Delta x, \Delta y, \Delta z)$ a punctului $P(x,y,z)$ se obține punctul $P'(x',y',z')$ după formula:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} + \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

... Transformări geometrice uzuale

b) Rotație

Rotația în jurul axei Oz cu unghiul α se obține astfel :

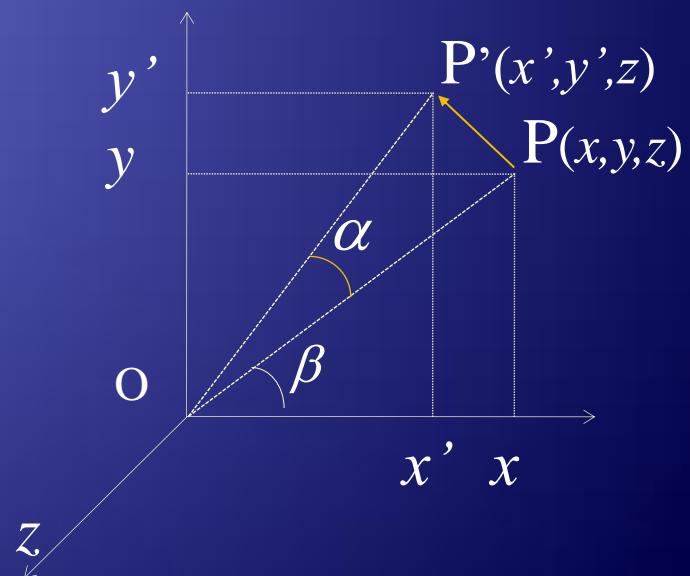
$$x' = OP' * \cos(\alpha + \beta) = OP \cos\alpha \cos\beta - OP \sin\alpha \sin\beta = x \cos\alpha - y \sin\alpha ,$$

$$y' = OP' * \sin(\alpha + \beta) = OP \sin\alpha \cos\beta + OP \cos\alpha \sin\beta = x \sin\alpha + y \cos\alpha , \quad \text{iar}$$

$$z' = z .$$

De aici rezultă :

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



Analog se deduc și formulele de rotație în jurul axelor Ox respectiv Oy .

... Transformări geometrice uzuale

c) Scalare

În funcție de factorul de scalare,

- 1) imaginea se *dilată* (factor de scalare supraunitar) sau
- 2) se *contractă* (factor subunitar) astfel:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = f_g * \begin{pmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & f_z \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

- ◆ f_g este un factor de scalare general,
- ◆ f_x, f_y, f_z sunt factorii de scalare pe cele trei axe.

... Transformări geometrice uzuale

d) Simetrie

Dând valori coeficienților S_x , S_y și S_z din mulțimea { -1, 0, 1 } se vor obține simetriile față de origine, axele de coordonate și planurile determinate de acestea.

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

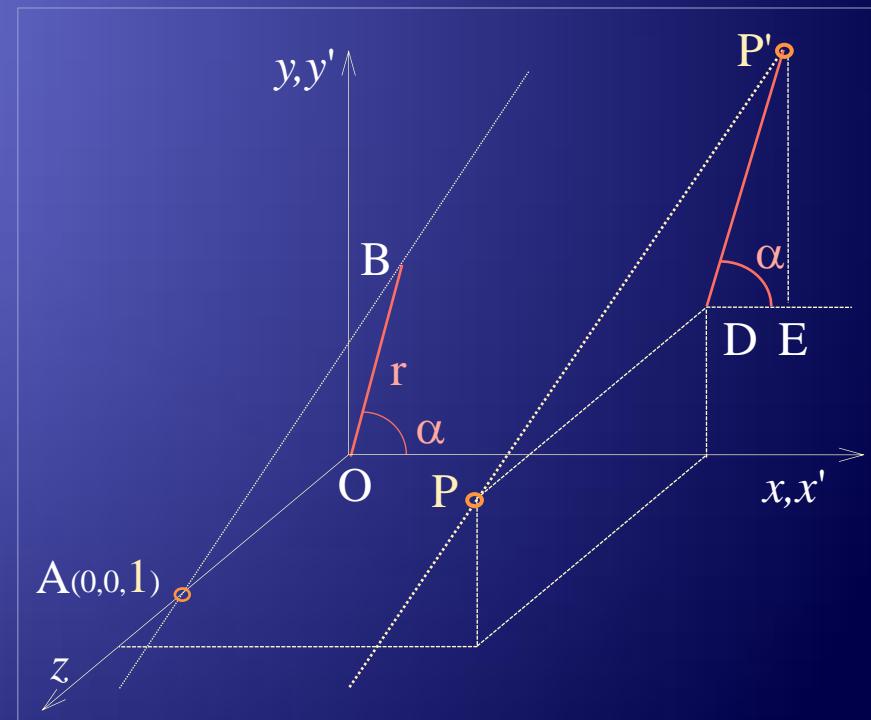
e) Proiecție

Pentru a putea reprezenta pe ecran obiecte tridimensionale vom proiecta mai întâi punctele din R3 în planul real, apoi prin transformările de fereastră (u și v prezentate în primul capitol) vom face trecerea din R2 în fereastra ecran. În cele ce urmează vom prezenta formulele de calcul pentru proiecția paralelă (sau cilindrică) și proiecția perspectivă (sau conică).

Pentru *proiecția paralelă* a un punct oarecare $P(x,y,z)$ va trebui să calculăm coordonatele x' și y' ale proiecției acestuia după direcția precizată (prin r și α). Se poate observa că triunghiul AOB este asemenea cu triunghiul PDP', de unde rezultă că $1/r = z/DP'$, deci $DP' = r \cdot z$. Deoarece

$$\begin{cases} x' = x + DP' \cdot \cos \alpha \text{ și} \\ y' = y + DP' \cdot \sin \alpha \end{cases} \quad \text{rezultă :}$$

$$\begin{cases} x' = x + r \cdot z \cdot \cos \alpha \quad \text{iar} \\ y' = y + r \cdot z \cdot \sin \alpha. \end{cases} \quad // PrX \quad // PrY$$



e) Proiecție ...

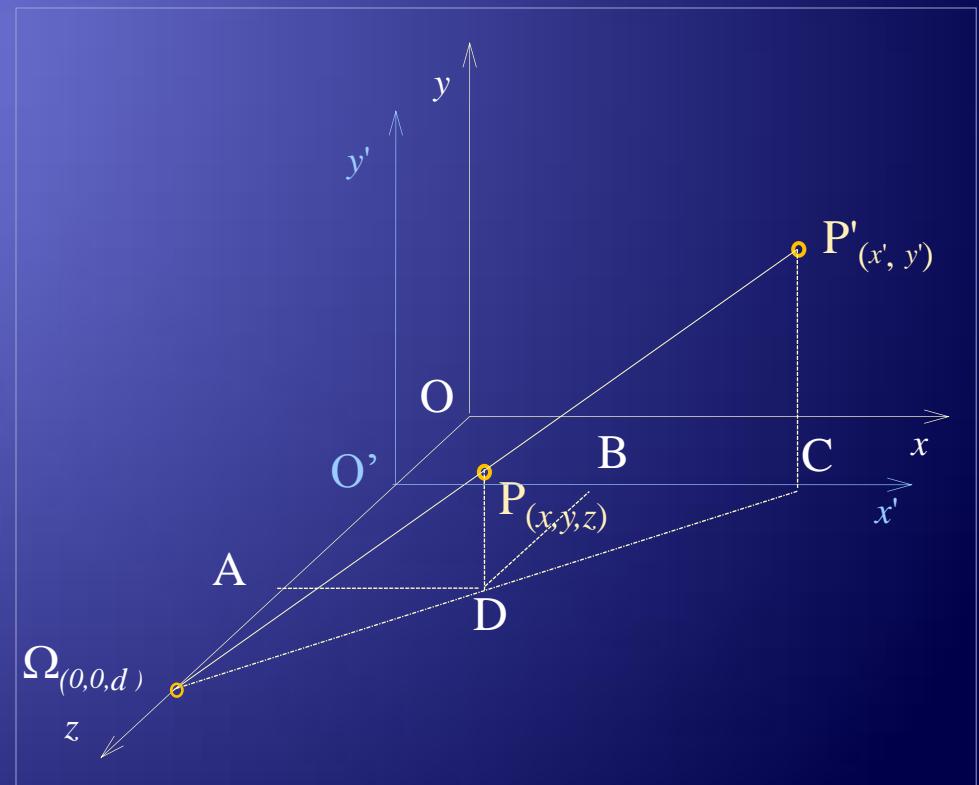
Pentru *proiecția conică* (definită prin distanța la care se află observatorul $\Omega(0,0,d)$ notată cu $d=O\Omega$ și distanța la care se află planul de proiecție notată cu $q=OO'$) observăm că:

$$\Delta\Omega AD \sim \Delta\Omega O'C \sim \Delta\Omega PD \sim \Delta\Omega P'C , \quad \text{de unde rezultă că:}$$

$$x / x' = y / y' = (d-z) / (d-q).$$

Deducem că:

$$\begin{cases} x' = x \cdot (d-q) / (d-z), & \text{iar } // PrX \\ y' = y \cdot (d-q) / (d-z). & // PrY \end{cases}$$



Reprezentarea curbelor, suprafețelor și corpurilor

Reprezentarea unei curbe se poate realiza prin unirea proiecțiilor unui sistem de puncte de pe aceasta, astfel:

```
double x (double t) { return Math.Cos(t); }          // x=f(t); y=g(t); z=h(t)
double y (double t) { return Math.Sin(t); }
double z (double t) { return           t/25; }
```

```
int      u1, v1, u2, v2;                           // ViewPort
double   a, b, c, d;                             // Window
double   Raza, Alfa;                            // Pr. Par.
int      Lu, Lv;                                // Lpr;
```

```
int u(double x) { return (int)((x - a) / (b - a) * (u2 - u1) + u1); }
int v(double y) { return (int)((y - d) / (c - d) * (v2 - v1) + v1); }

void ViewPort(int x1, int y1, int x2, int y2) { u1 = x1; v1 = y1; u2 = x2; v2 = y2; }
void Window(double x1, double y1, double x2, double y2) { a = x1; d = y1; b = x2; c = y2; }
```

... Reprezentarea curbelor, suprafețelor și corpurilor

```
...
void DefPr (double r, double a) { Raza = r; Alfa = a; } // r=1; a=0.8; // = Pi/4
double PrX (double x, double z) { return x+Raza*z*Math.Cos(Alfa); }
double PrY (double y, double z) { return y+Raza*z*Math.Sin(Alfa); }

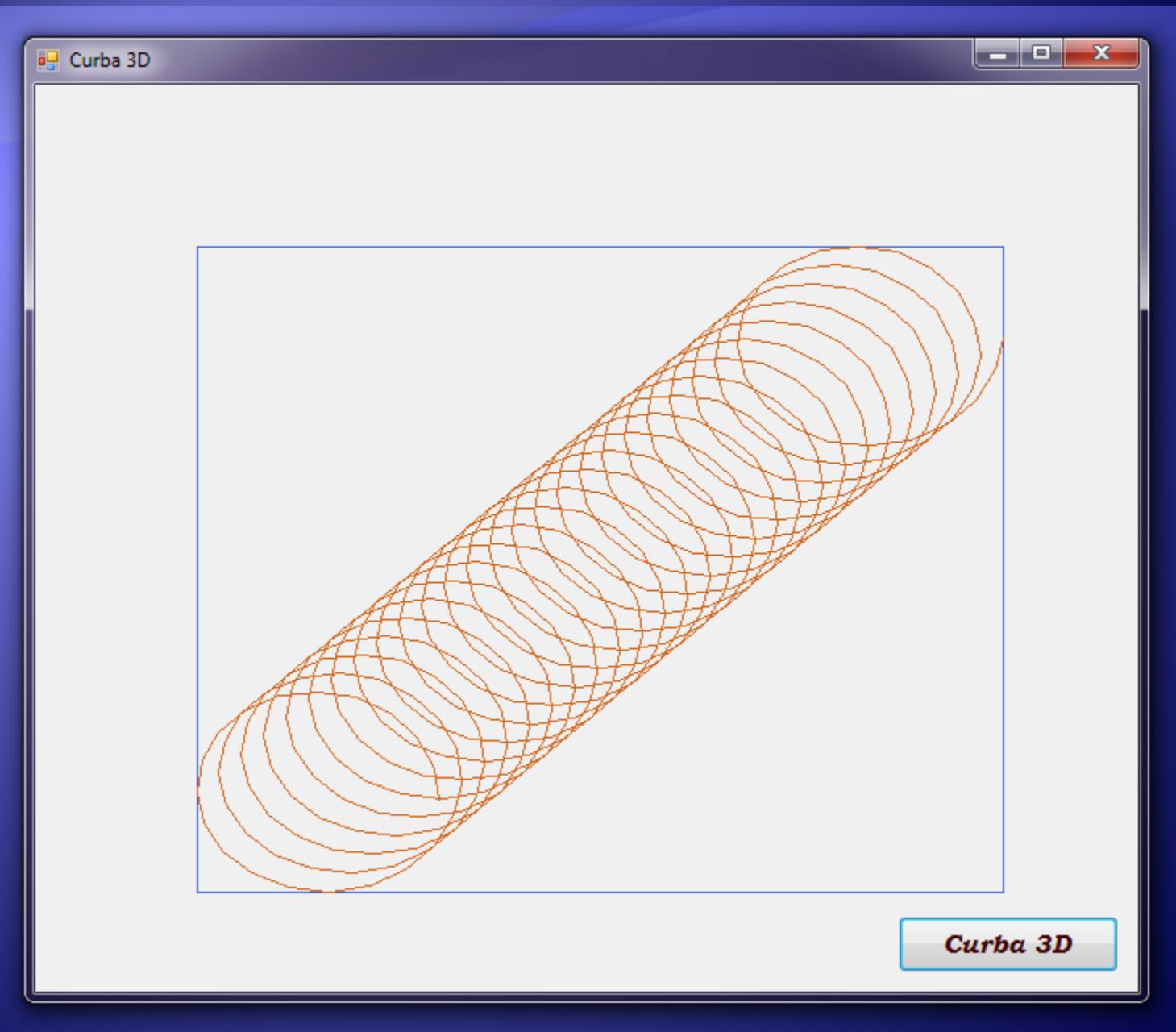
void MoveTo (int u1, int v1) { Lu = u1; Lv = v1; }
void LineTo (int u1, int v1, System.Drawing.Graphics Gr, System.Drawing.Pen Pen)
{ Gr.DrawLine(Pen, Lu, Lv, u1, v1); Lu = u1; Lv = v1; }
...
```

```
...
ViewPort (100, 100, 600, 500);
DefPr(1, 3.14/4); int n=500;
double t1=0, t2=50*3.1416; // Domeniul de definiție
...
```

... Reprezentarea curbelor, suprafețelor și corpurilor

```
double a=PrX(x(t1),z(t1)), b=PrX(x(t1),z(t1)),    // Determinarea ferestrei reale
       c=PrY(y(t1),z(t1)), d=PrY(y(t1),z(t1));
for (int i=1; i<=n; i++) {
    double t=(t2-t1)/n*i+t1;
    double Xp=PrX(x(t),z(t)), Yp=PrY(y(t),z(t));
    if (Xp<a) a=Xp; else if (Xp>b) b=Xp;
    if (Yp<c) c=Yp; else if (Yp>d) d=Yp;
}
Window (a,d,b,c);
MoveTo(u(PrX(x(t1),z(t1))),v(PrY(y(t1),z(t1))));      // Desenarea Curbii
for (int i=1; i<=n; i++) {
    double t=(t2-t1)/n*i+t1;
    LineTo(u(PrX(x(t),z(t))),v(PrY(y(t),z(t))),formGraphics, myPen);
}
```

... Reprezentarea curbelor, suprafețelor și corpurilor



Reprezentarea curbelor, suprafețelor și corpurilor

Pentru a reprezenta o suprafață definită de o funcție $z : [a,b] \times [c,d] \rightarrow \mathbb{R}$, va fi necesară o proiecție a punctelor din \mathbb{R}^3 în planul XOY. Vom considera o proiecție paralelă după o direcție precizată prin cele două elemente r și α .

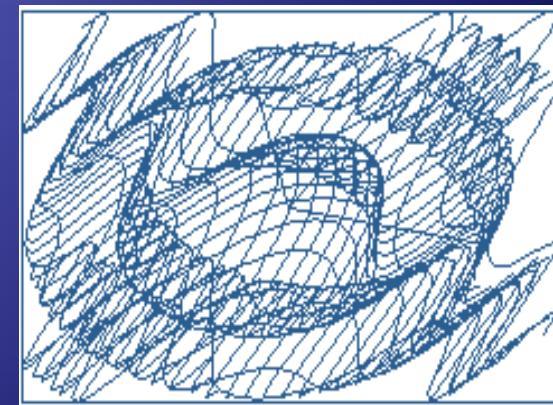
Ca exemplu, vom desena suprafața descrisă de funcția:

$z : [-\pi, \pi] \times [-\pi, \pi] \rightarrow \mathbb{R}$, $z(x,y) = \sin x^2 + \cos y^2$, aplicând o proiecție paralelă de direcție $(r, \alpha) = (1, \pi/4)$.

Trasarea suprafeței se reduce la o problemă plană. Pentru un punct x fixat, iar y variabil din intervalul $[c,d]$ se trasează graficul proiecției corespunzătoare punctelor $P(x,y,z(x,y))$.

Acest lucru se realizează pentru mai multe puncte x alese din intervalul $[a,b]$ ($n+1$ puncte echidistante).

Același lucru se realizează pentru y fix și x variabil, realizând astfel o imagine sub forma unei *plase* (ca în figura alăturată):



... Reprezentarea curbelor, suprafețelor și corpuriilor

Practic se reprezintă două siruri de curbe $z(x_i, y)$ și $z(x, y_j)$, unde :

$$x_i = a + i^*(b-a)/n \quad (i=0,1,\dots,n),$$

$$y_j = c + j^*(d-c)/m \quad (j=0,1,\dots,m), \quad \text{iar } n \text{ și } m \text{ reprezintă finețea rețelei.}$$

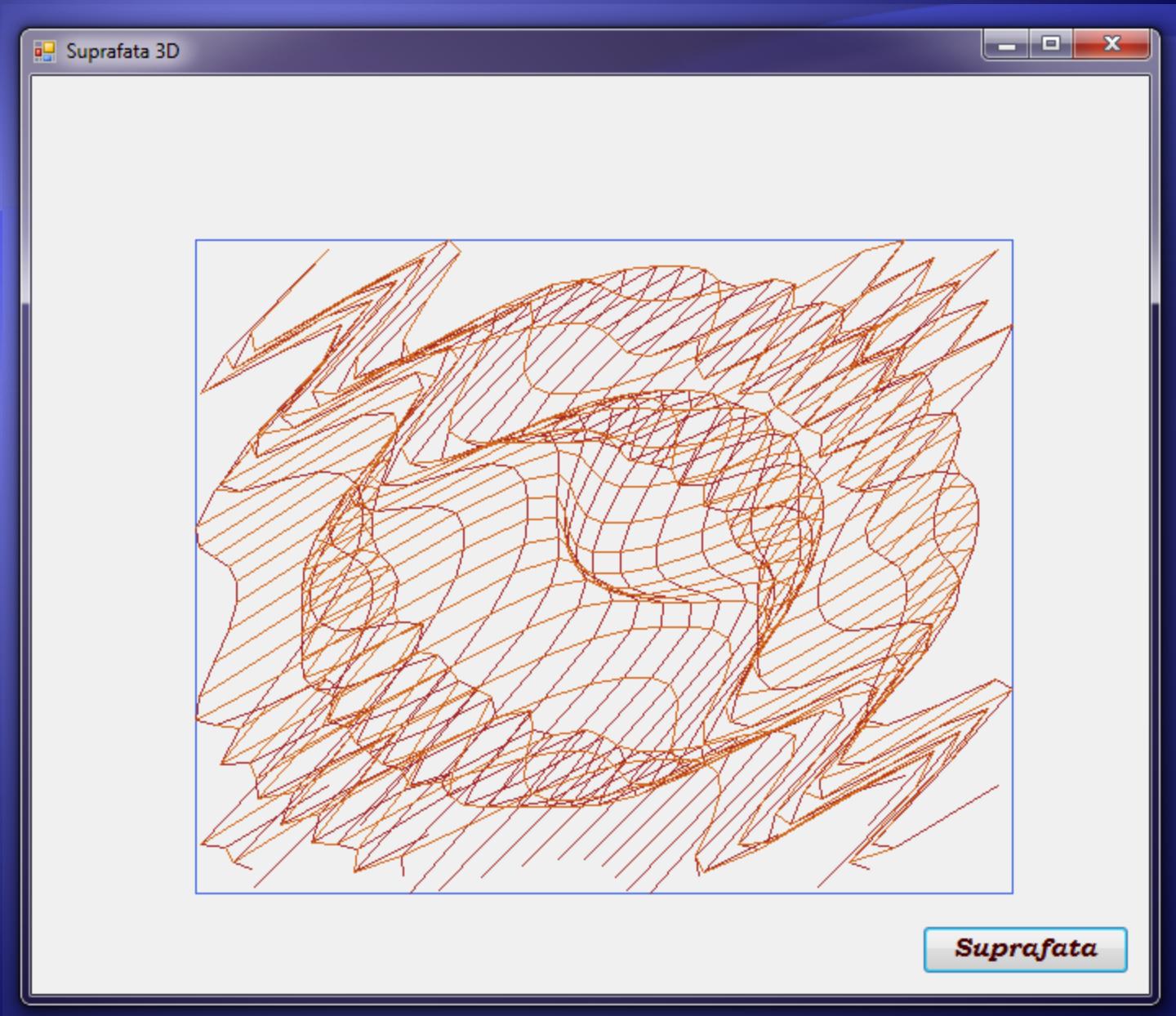
```
...
ViewPort (100, 100, 600, 500);
double Pi=3.1416; DefPr(1, 3.14/4); int n=25, m=25;
double a=-Pi, b=Pi, c=-Pi, d=Pi; // Domeniul de definiție
double Pas_x=(b-a)/n, Pas_y=(d-c)/m;
double Wl=PrX(a,z(a,c)), Wr=PrX(a,z(a,c)); // Max,Min / x',y' (Window)
double Wd=PrY(c,z(a,c)), Wt=PrY(c,z(a,c));
for (double x=a; x<=b; x+=Pas_x)
    for (double y=c; y<=d; y+=Pas_y) {
        double Ux=PrX(x,z(x,y)), Uy=PrY(y,z(x,y));
        if (Ux<Wl) Wl=Ux; else if (Ux>Wr) Wr=Ux;
        if (Uy<Wd) Wd=Uy; else if (Uy>Wt) Wt=Uy;
    }
Window (Wl, Wt, Wr, Wd);
```

... Reprezentarea curbelor, Suprafetelor și corpurilor

```
...
for (double x = a; x <= b; x += Pas_x)    {
    double y = c;      MoveTo(u(PrX(x, z(x, y))), v(PrY(y, z(x, y))));
    for (y = c+Pas_y; y <= d; y += Pas_y)
        LineTo(u(PrX(x, z(x, y))), v(PrY(y, z(x, y))), formGraphics, myPen);
}
```

```
for (double y = c + Pas_y; y <= d; y += Pas_y)    {
    double x = a;      MoveTo(u(PrX(x, z(x, y))), v(PrY(y, z(x, y))));
    for (x = a; x <= b; x += Pas_x)
        LineTo(u(PrX(x, z(x, y))), v(PrY(y, z(x, y))), formGraphics, myPen);
}
...
```

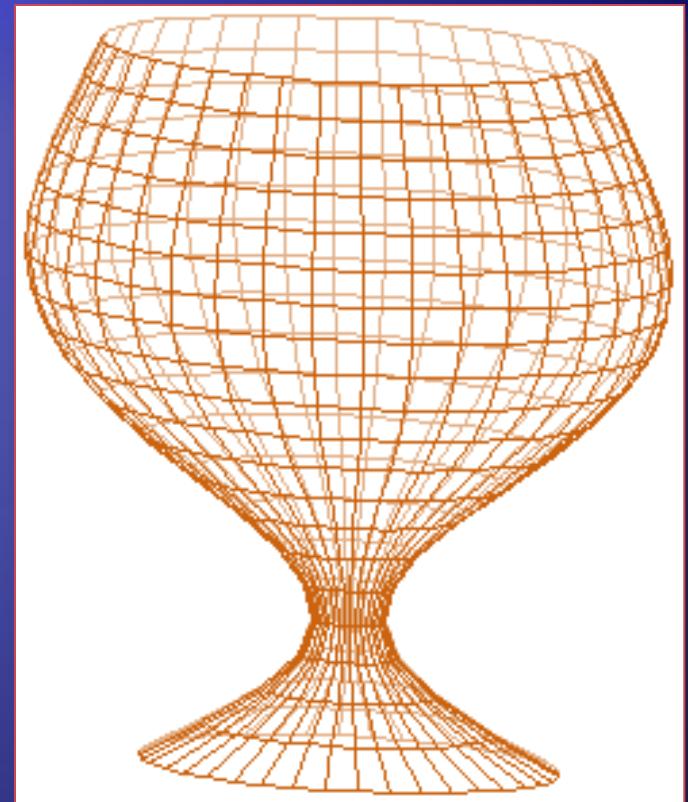
... Reprezentarea curbelor, Suprafețelor și corpurilor



Un alt exemplu (pe care il propunem ca tema) constă în construirea și reprezentarea grafică a unei *suprafețe de rotație*.

Un alt exemplu (pe care îl propunem ca temă) constă în construirea și reprezentarea grafică a unei suprafețe de rotație.

Suprafața reprezentată în figura alaturată am generat-o plecând de la funcția $z(y) = \sin y^2$, al cărei grafic se rotește în jurul axei Oy , apoi se reprezintă segmentele ce unesc punctele obținute prin rotire. În felul acesta sunt reprezentate două sisteme de curbe de tip meridiane și paralele aşa cum se poate vedea în figura alaturată.



Reprezentarea curbelor, suprafețelor și Corpurilor

Reprezentarea pe ecran a unui corp modelat prin muchiile sale (date într-un fișier text) este realizată prin desenarea proiecțiilor acestora.

```
int Tip; double Raza, Alfa;           // Pr. Par.=1, Perp.=2
class muchie { public int st, dr; }     // Pot fi și caracteristici: Culoare, TipLinie...
class varf {
    public double x, y, z;
    public varf(int X, int Y, int Z) { x = X; y = Y; z = Z; }
}
```

```
void DefPr(double r, double a) { Raza = r; Alfa = a; }      // r=1; a=0.8; // = Pi/4
double PrX(double x, double z) { return x + Raza * z * Math.Cos(Alfa); }
double PrY(double y, double z) { return y + Raza * z * Math.Sin(Alfa); }

double Px(varf P) { return PrX(P.x, P.z); }
double Py(varf P) { return PrY(P.y, P.z); }
```

... Reprezentarea curbelor, suprafețelor și Corpurilor

```
...
DefPr(Raza, Alfa); // 1=Par(r,a), 2=Persp.(d,q);
a = b = Px(V[1]); c = d = Py(V[1]);
for (int i = 2; i <= n; i++)
{
    double px = Px(V[i]); if (px < a) a = px; else if (px > b) b = px;
    double py = Py(V[i]); if (py < c) c = py; else if (py > d) d = py; }

Window(a, d, b, c); // Fereastra Reală
for (int j = 1; j <= m; j++)
    formGraphics.DrawLine(myPen, u(Px(V[M[j].st])), v(Py(V[M[j].st])),
                          u(Px(V[M[j].dr])), v(Py(V[M[j].dr])));

...
```

... Reprezentarea curbelor, suprafețelor și Corpurilor

Fișierul text conține informațiile:

Pentru cubul de mai jos, datele sunt :

n
 $x_1 \ y_1 \ z_1$
 $x_2 \ y_2 \ z_2$
...
 $x_i \ y_i \ z_i$
...
 $x_n \ y_n \ z_n$

*Lista de vârfuri dată explicit
 $P_i (x_i, y_i, z_i), i=1, n$*

m
 $s_1 \ d_1 \ c_1$
 $s_2 \ d_2 \ c_2$
...
 $s_j \ d_j \ c_j$
...
 $s_m \ d_m \ c_m$

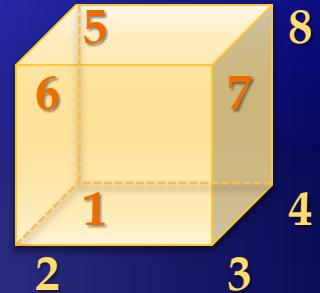
*Lista de muchii dată implicit
prin indici de puncte:
 $S_j (s_j, d_j, c_j), j=1, m$*

$Tp \ r/d \ \alpha/q$

Definirea proiecției

8
0 0 0
1 0 0
1 1 0
0 1 0
0 0 1
1 0 1
1 1 1
0 1 1

12
1 2 1
2 3 1
3 4 1
4 1 1
5 6 2
6 7 2
7 8 2
8 5 2
1 5 3
2 6 3
3 7 3
4 8 3
1 1.0 0.4

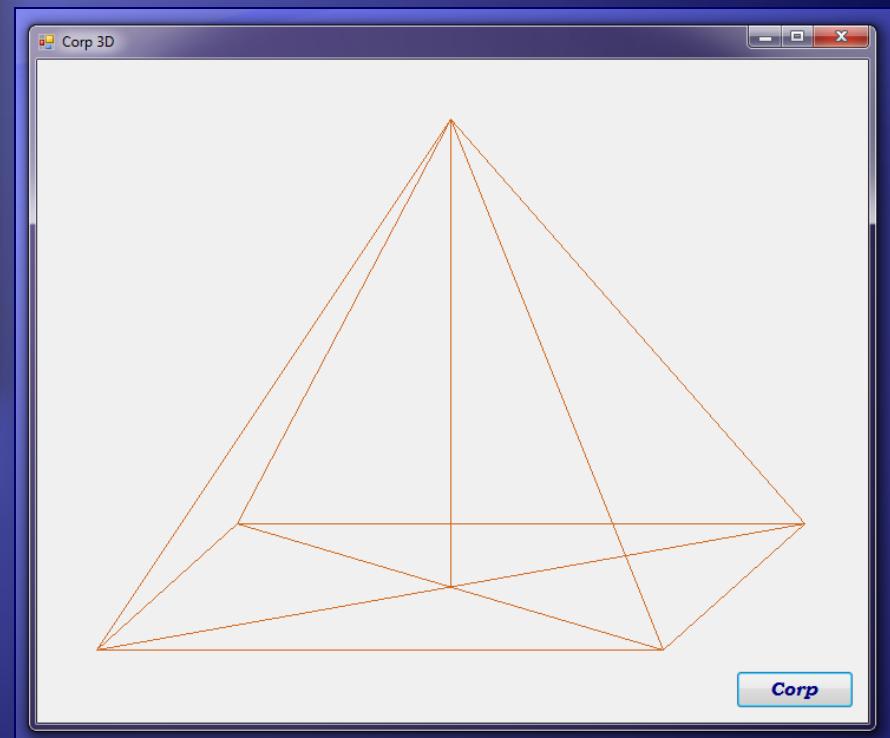
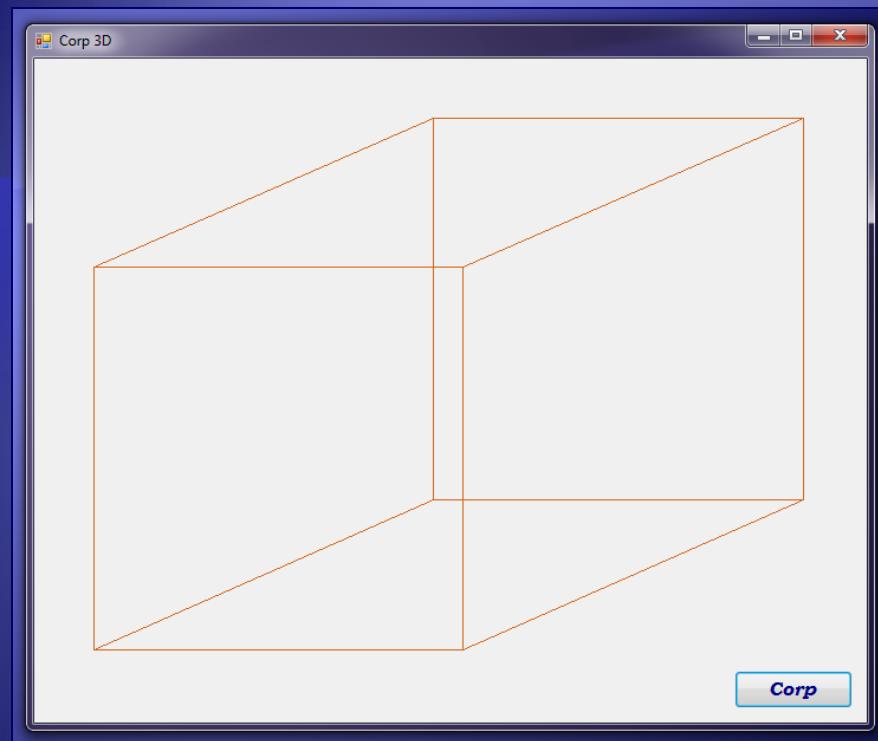


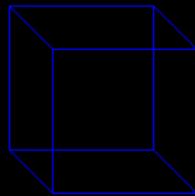
... Reprezentarea curbelor, suprafețelor și Corpurilor

Fișierul text conține următoarele date :

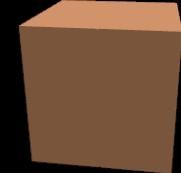
Cub.Txt			Piramida.Txt		
Listă Vârfuri	Listă Muchii	Proiecție	Listă Vârfuri	Listă Muchii	Proiecție
8	12	1 1.0 0.4	6	11	1 3.5 1.5
0 0 0	1 2 1		-50 -50 -400	1 2	
1 0 0	2 3 1		-50 50 -400	2 3	
1 1 0	3 4 1		50 50 -400	3 4	
0 1 0	4 1 1		50 -50 -400	4 1	
0 0 1	5 6 2		0 0 900	1 5	
1 0 1	6 7 2		0 0 -400	2 5	
1 1 1	7 8 2			3 5	
0 1 1	8 5 2			4 5	
	1 5 3			1 3	
	2 6 3			2 4	
	3 7 3			5 6	
	4 8 3				

... Reprezentarea curbelor, suprafețelor și Corpurilor

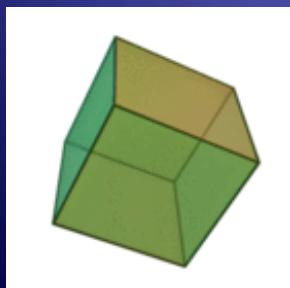
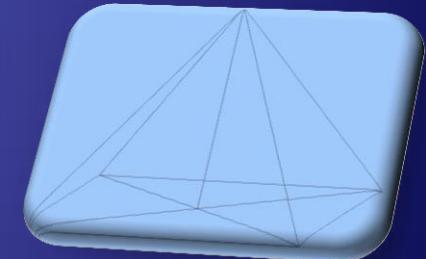
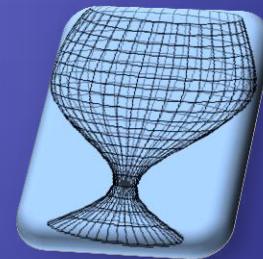
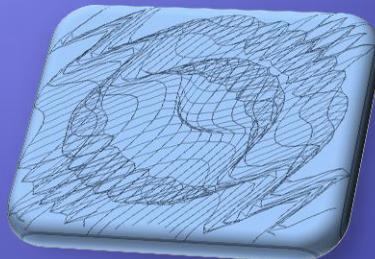
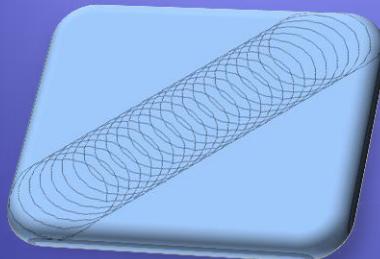




Temă



Să se reprezinte cel puțin un obiect 3D:
curbă, suprafață [de rotație], corp <prin muchii/fete>.



Success!

