

# 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
  - Texturi

# *Creșterea realismului imaginilor tridimensionale*

Următoarele cursuri prezintă câteva metode de îmbunătățire a imaginilor în sensul apropierea calității lor de imaginile reale.

Dintre aceste metode prezentate în literatura de specialitate cum ar fi:

- a) *eliminarea suprafețelor și muchiilor acoperite* pentru extragerea elementelor de frontieră ascunse,
  - b) *perspectiva* pentru informațiile de profunzime,
  - c) *proiecțiile dinamice* pentru reprezentarea obiectelor în mișcare,
  - d) *indici de intensitate sau variația de culoare* utilizate pentru modificarea culorilor din adâncime,
  - e) ***texturi*** și detalii de suprafață pentru reprezentarea microstructurilor fețelor,
  - f) *secționarea* cu un plan frontal utilizată la vizualizarea interiorului obiectului,
  - g) *iluminarea curpurilor* prin *utilizarea luminilor și umbrelor* și
  - h) *stereografie* pentru redarea în relief a obiectelor tridimensionale,
- am ales doar câteva pe care le-am considerat mai importante.

*... Creșterea realismului imaginilor tridimensionale*

*1. Eliminarea suprafețelor acoperite*

*2. Texturi*

*3. Lumină și umbră*

*4. Stereografie*

## 2. *Texturi*

*Utilizarea texturilor* în grafica tridimensională are un rol important în creșterea realismului imaginilor (mai ales dacă acestea prezintă mici imperfecțiuni, adică defecte intenționat introduse), deoarece aceste detalii de pe suprafața obiectelor dau multe informații (*material, poziție, dimensiuni, etc.*) despre corpul reprezentat.

Există trei categorii de texturi și anume:

- a) *Constance* ca mărime și orientare pe suprafața corpului,
- b) *Variabile* ca mărime și orientare în funcție de poziția fețelor.
- c) *Neregulate* - aleatoare ca mărime și orientare (*fractali*).

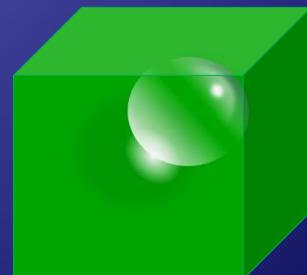
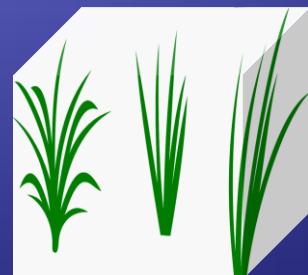
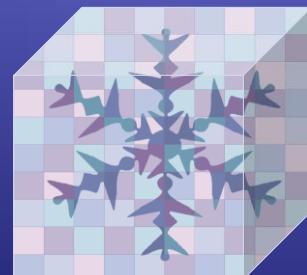
## ... 2. *Texturi* ~ *Texturi constante*

### a) *Texturile constante*

Acestea se utilizează atât în grafica 2-D cât și în grafica 3-D pentru tapetarea fondurilor scenelor.

Acestea se pot defini printr-o matrice de culori sau în sistem vectorial prin coodonate relative ale capitelor segmentelor descrise.

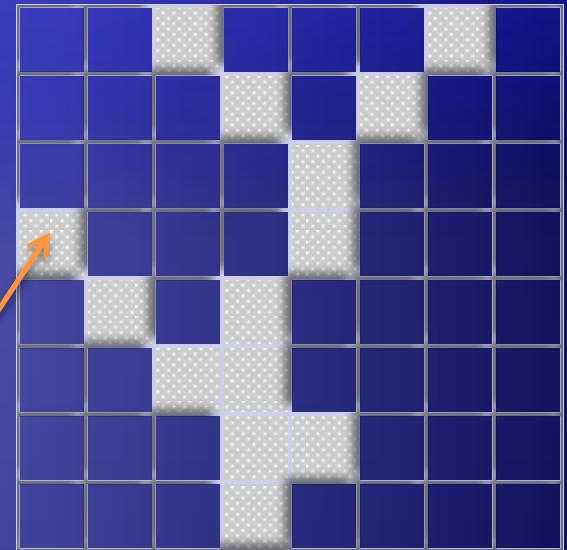
Există biblioteci de texturi caracteristice diferitelor obiecte sau materiale cum ar fi: *fagure, lemn, parchet, sticlă, apă, iarbă*, etc.



## ... 2. Texturi ~ Texturi constante

Aplicarea unei texturi se poate face la umplere, unde un punct  $P_{ij}$  se va colora ținând cont de o textură definită prin matricea  $T$  cu  $m$  linii și  $n$  coloane numerotate de la 0 la  $m-1$  respectiv  $n-1$  în culoarea  $T_{i \bmod m, j \bmod n}$ .

$$T_{ij} = \text{Textel}$$



Aplicarea unei texturi se poate face la umplere, unde un punct  $P_{ij}$  se va colora ținând cont de o textură definită prin matricea  $T$  cu  $m$  linii și  $n$  coloane numerotate de la 0 la  $m-1$  respectiv  $n-1$  în culoarea  $T_{i \bmod m, j \bmod n}$ .

## ... 2. *Texturi* ~ *Texturi variabile*

### b) *Texturile variabile*

Un exemplu des utilizat de texturi variabile îl constituie aplicarea textelor pe fețele corpurilor (simboluri scrise pe suprafața obiectelor).

Pentru aceasta este nevoie de o *digitizare* a caracterelor prin segmente sau prin linii poligonale închise.

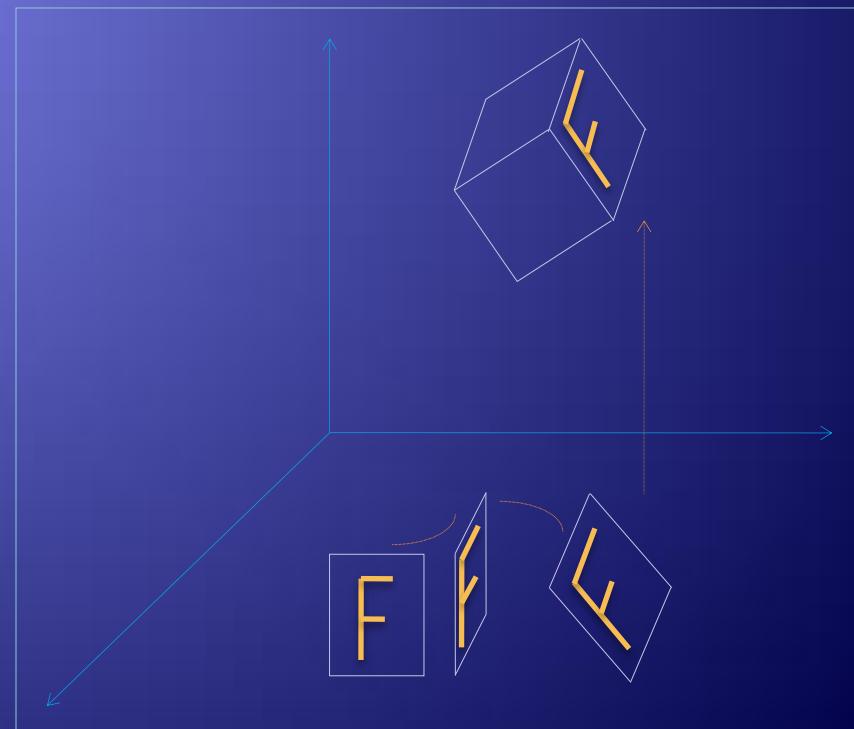
Există o adevărată industrie de fonturi care crează biblioteci de caractere .



## ... 2. *Texturi* ~ *Texturi variabile*

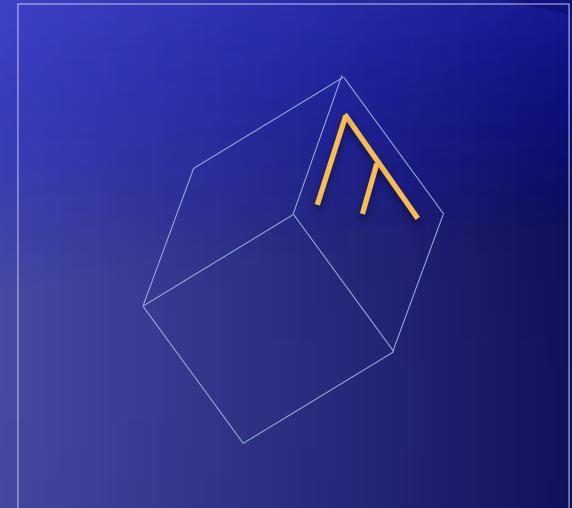
Aplicarea pe față dorită a unei texturi variabile se realizează în trei etape:

- Se aduce caracterul pe față dorită prin două rotații (într-un plan paralel cu fața pe care se dorește aplicarea) apoi printr-o translație;
- Se proiectează punctele critice care descriu caracterul odată cu obiectul;
- Se construiește textura unind punctele critice sau prin umplere.



## *Observație*

La caracterele nesimetrice trebuie avută în vedere și normala la plan, deoarece caracterele pot fi aplicate invers (ca în figura alăturată).

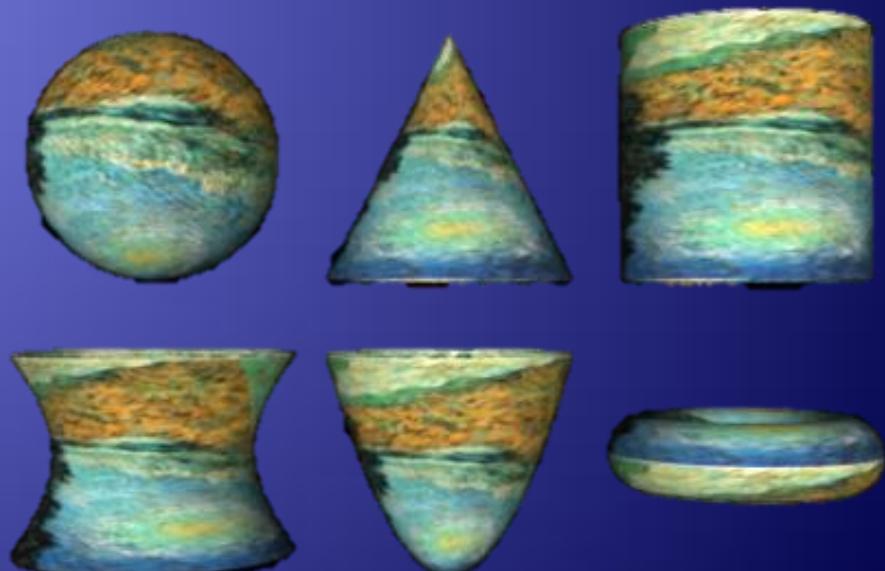


## Aplicarea texturilor pe suprafețe

$$P(x,y,z) \xrightarrow{f} T_{ij} \quad f = ?$$

### Aplicarea texturilor

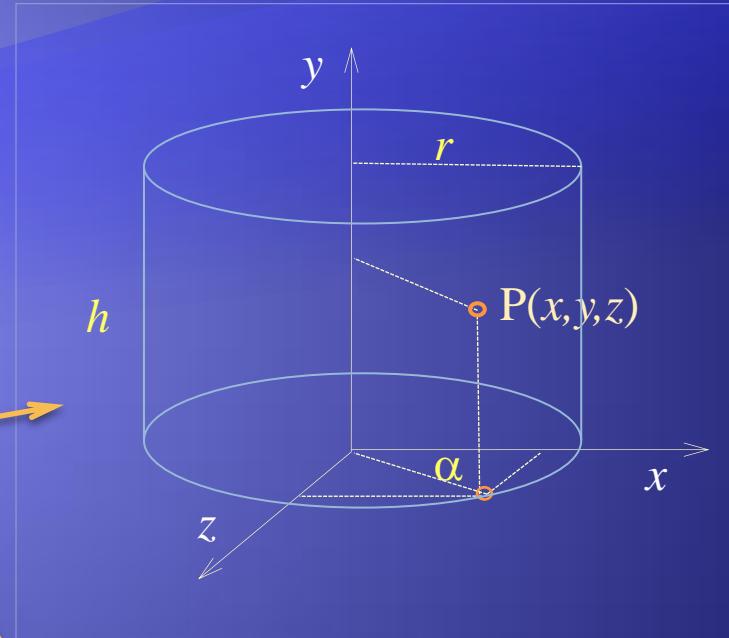
- ❖ pe suprafețe plane
- ❖ pe cilindru ( $r,h$ )
- ❖ pe sferă ( $r$ )
- ❖ pe alte obiecte



## ... Aplicarea texturilor pe suprafețe

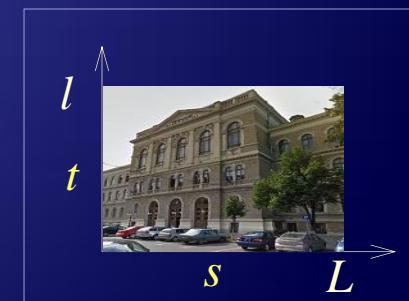
### Aplicarea texturilor

- ❖ pe suprafețe plane
- ❖ pe cilindru  $(r,h)$  :
- ❖ pe sferă  $(r)$
- ❖ pe alte obiecte



$$\begin{cases} x = r \cos \alpha \\ z = r \sin \alpha \\ y = y \end{cases}$$

$$\begin{cases} s = \alpha * L / \pi \\ t = y * l / h \end{cases}$$



... Aplicarea texturilor pe suprafețe

- ❖ ...
- ❖ pe cilindru  $(r,h)$  :
- ❖ pe sferă  $(r)$
- ❖ ...



```
int L = 399, l = 399; // Textura (Lung, lat.)
```

```
double Rc = 1, h = 2; // Cilindru (Rc, h)
```

```
double x(double Rc, double Alf) { return Rc * Math.Cos(Alf); } // x = r * cos (Alfa); r=Rc=Raza Cil.
```

```
double z(double Rc, double Alf) { return Rc * Math.Sin(Alf); } // z = r * sin (Alfa)
```

```
int s(double Alf, double L) { return (int)(L-Alf * L / Pi ); } // Textura[s,t] din [0,L]x[0,l]
```

```
int t(double y, int l, double h) { return (int)(l - y * l / h ); }
```

... Aplicarea texturilor pe suprafețe

- ❖ ...
- ❖ pe cilindru( $r,h$ ):
- ❖ pe sferă ( $r$ )
- ❖ ...

```

ViewPort (100, 50, 300, 350);    DefPr (1, 3.14 / 4);
double a = b = PrX( $x(Rc, 0)$ ,  $z(Rc, 0)$ ),    // Determinarea ferestrei reale
      c = d = PrY(0,           $z(Rc, 0)$ );
for (double al = 0; al < Pi; al += Pi / L)      // Alfa din [0,Pi]
for (double y = 0; y < h; y += h / l)           // y din [0, h]
{
    double Cx =  $x(Rc, al)$ , Cy = y, Cz =  $z(Rc, al)$ ; // Pct. pe Cil.(Cx, Cy, Cz)
    double Xp = PrX(Cx, Cz), Yp = PrY(Cy, Cz);
    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);

```

```

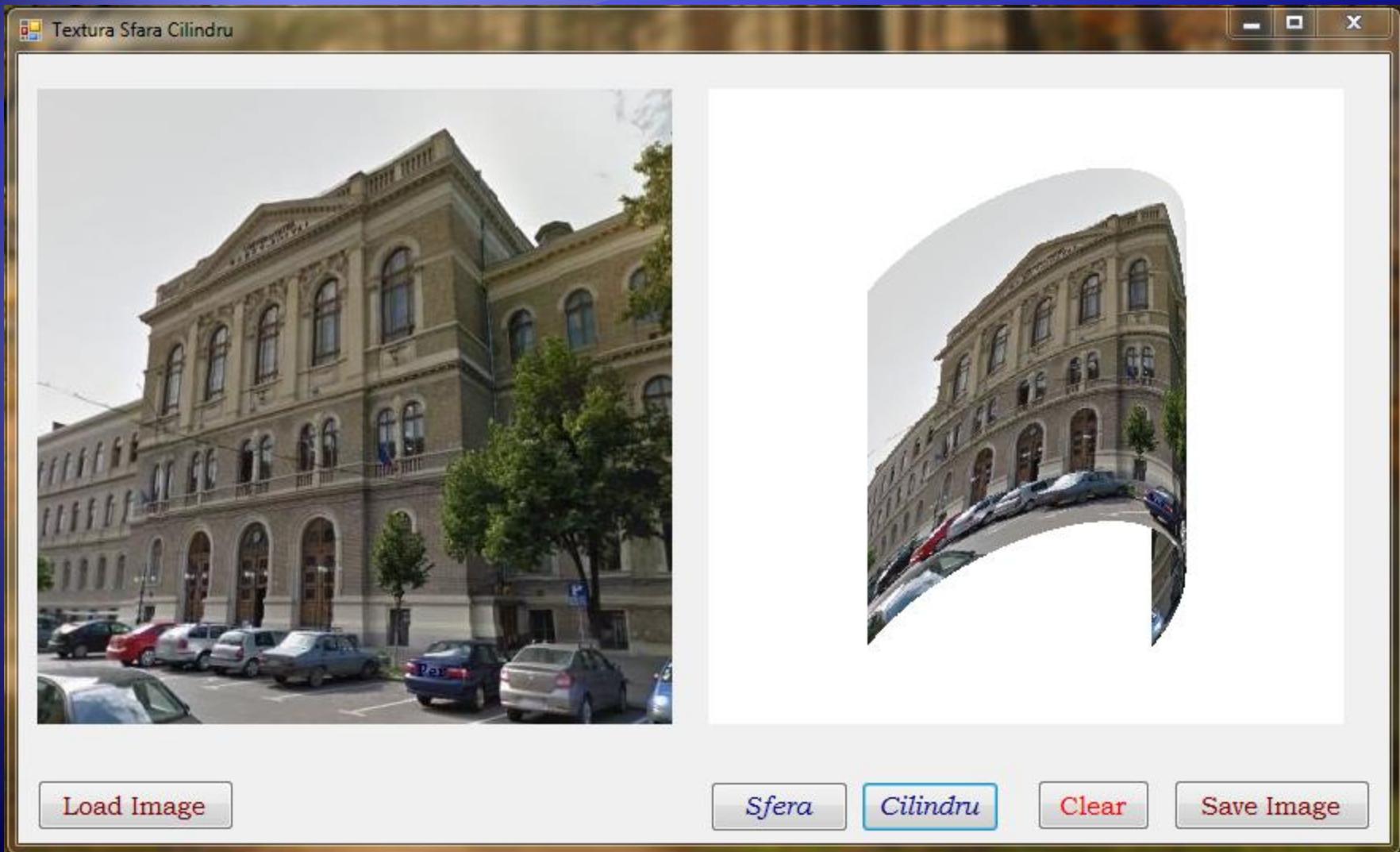
for (double al = 0; al < Pi; al += Pi / L)      // Alfa din [0,Pi]
    for (double y = 0; y < h; y += h / l)           // y din [0, h]
{
    double Cx =  $x(Rc, al)$ , Cy = y, Cz =  $z(Rc, al)$ ; // Punct pe Cil.(Cx, Cy, Cz)
    double Px = PrX(Cx, Cz), Py = PrY(Cy, Cz);
    Im2.SetPixel(u(Px), v(Py), Im1.GetPixel(s(al, L), t(y, l, h)));
}

```

... Aplicarea texturilor pe suprafețe

... 2. Texturi ~ Texturi variabile

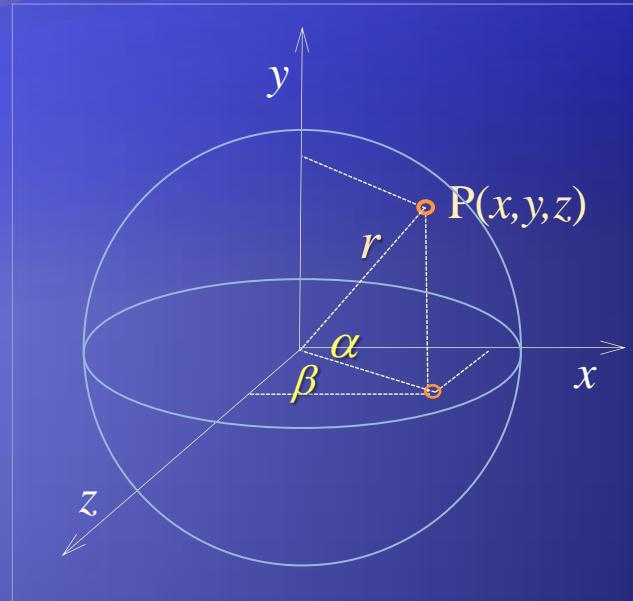
❖ pe cilindru( $r, h$ ):



... Aplicarea texturilor pe suprafețe

Aplicarea texturilor

- ❖ pe suprafețe plane
- ❖ pe cilindru  $(r, h)$  :
- ❖ pe sferă  $(r)$  :
- ❖ pe alte obiecte



$$\begin{cases} x = r \cos \alpha \sin \beta \\ y = r \sin \alpha \\ z = r \cos \alpha \cos \beta \end{cases}$$



... Aplicarea texturilor pe suprafețe

❖ pe sferă ( $r$ ):

```

double alfa(int j, int H) { double Pi = 3.141592;
    return -Pi * j / H + Pi / 2;
}

double beta(int i, int W) { double Pi = 3.141592;
    return -Pi * i / W + Pi;
}

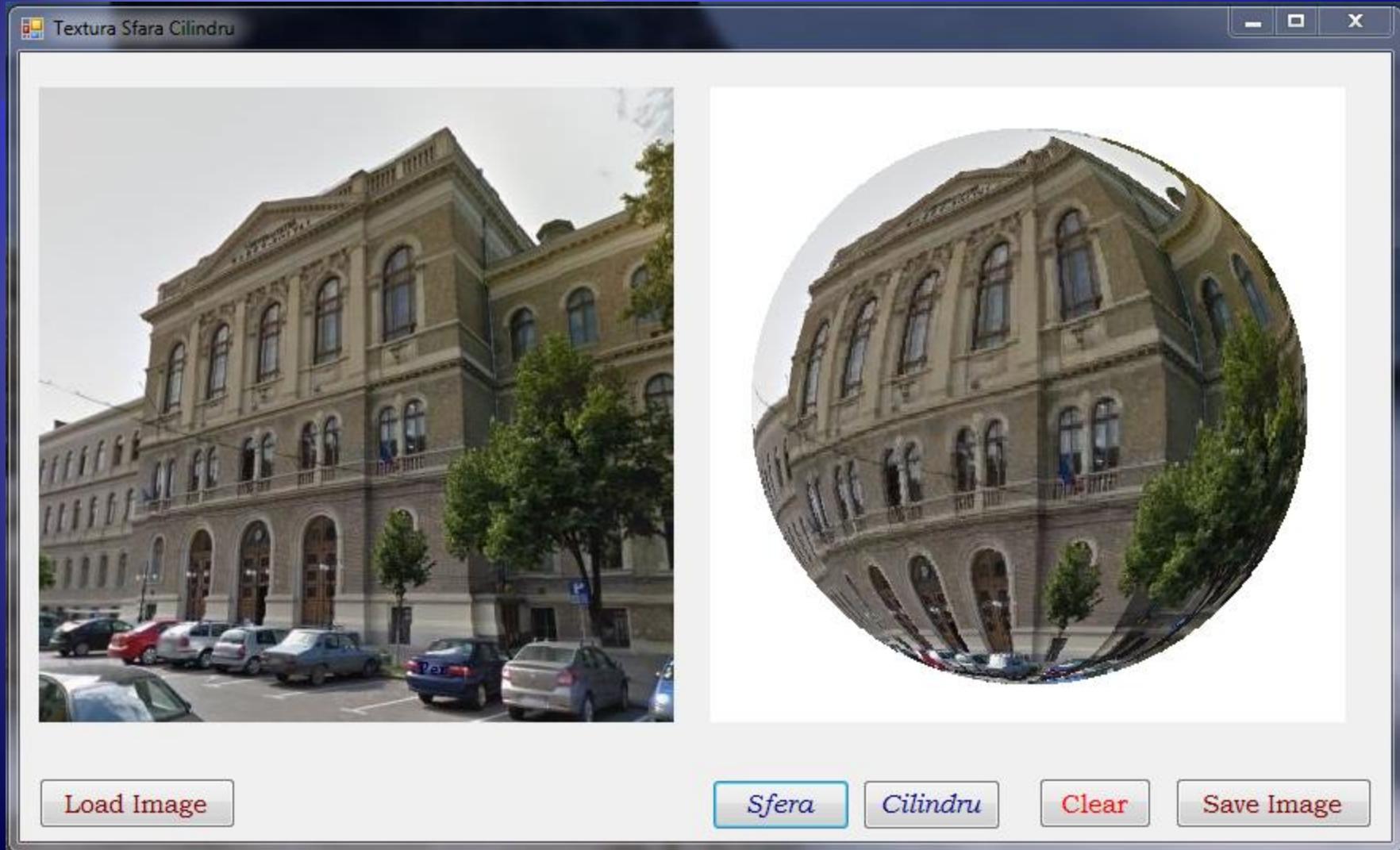
void PutPixel(Bitmap Im, int x, int y, Color c)
{
    Im.SetPixel(x, y, c); Im.SetPixel(x, y+1, c);
}

private void button2_Click(object sender, EventArgs e) // Sfera
{
    int Ri = 90, Rj = 185;
    for (int i = 0; i < Im1.Width; i++)
        for (int j = 0; j < Im1.Height; j++)
            PutPixel(Im2, Im1.Width/2 + (int)(Math.Cos(alfa(j,Im1.Height)) * Math.Cos(beta(i, Im1.Width)) * (Ri + j / 5)),
                     Im1.Height/2 - (int)(Math.Sin(alfa(j, Im1.Height)) * Rj), Im1.GetPixel(i, j));
    pictureBox2.Image = Im2; pictureBox2.Refresh();
}

```

... Aplicarea texturilor pe suprafețe

❖ pe sferă ( $r$ ):



... Aplicarea texturilor pe suprafețe

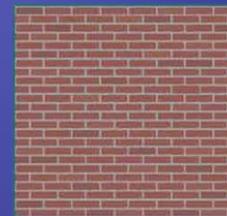
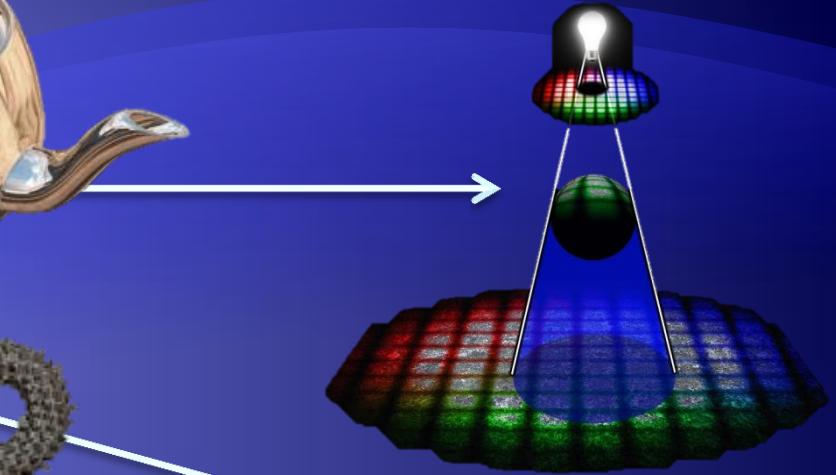


## Referințe

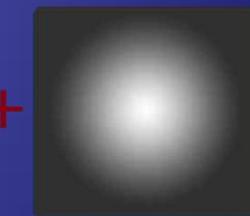
- ❖ [http://csclab.murraystate.edu/bob.pilgrim/515/texture\\_maps\\_project.html](http://csclab.murraystate.edu/bob.pilgrim/515/texture_maps_project.html)
- ❖ <http://www.flashandmath.com/advanced/sphere/ideas.html>
- ❖ <https://www.siggraph.org/education/materials/HyperGraph/mapping/spheretx.htm>
- ❖ <https://www.cs.cmu.edu/~fp/courses/graphics/pdf-color/10-texture.pdf>
- ❖ <http://www.cs.kent.edu/~zhao/acg13/lectures/IntroTextureMapping.pdf>
- ❖ [http://web.eecs.utk.edu/~huangj/cs456/notes/456\\_texturemap1.pdf](http://web.eecs.utk.edu/~huangj/cs456/notes/456_texturemap1.pdf)

## Tipuri de texturi

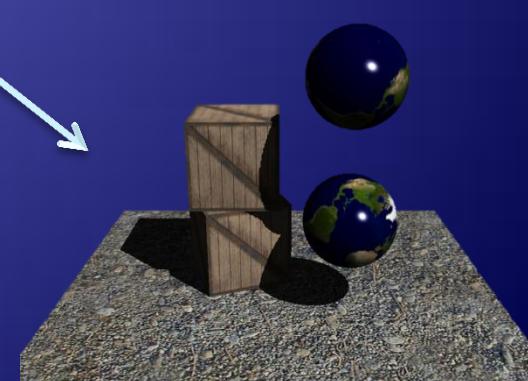
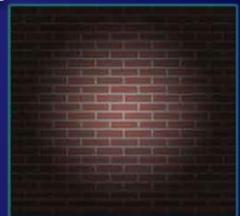
- ❖ de mediu
- ❖ proiective
- ❖ de iluminare
- ❖ de umbrire
- ❖ celulare
- ❖ combineate
- ❖ cu defecte (Bump)



+



=



## ... 2. *Texturi* ~ *Texturi neregulate*

### c) *Texturile neregulate*

Acstea texturi sunt utilizate în generarea aleatoare a unor forme de simulare a realității, de exemplu *arbori*, *nori*, *dune de nisip*, *valuri*, *lemn*, etc.

Realizarea unor astfel de texturi se poate face prin:

- Variația aleatoare a *direcției de iluminare*;
- Distorsionarea aleatoare a *normalei la plan*;
- *Structuri fractale*.

*Fractalii*, aceste variațiuni geometrice ciudate, pot fi:

- de *formă regulată* (generați prin repetarea unui motiv sau detaliu primar) sau
- de *formă neregulată* (definiți *probabilistic*).

## ... 2. *Texturi ~ Texturi neregulate*

Fractalii de formă regulată se definesc prin *curbe, suprafete, volume, funcții*, etc., sau printr-o *regulă de construcție*.

Fractalii sunt forme definite prin ecuații matematice, rezultând o figură geometrică, care poate fi divizată în părți, astfel încât fiecare dintre acestea să fie o copie miniaturală a întregului.

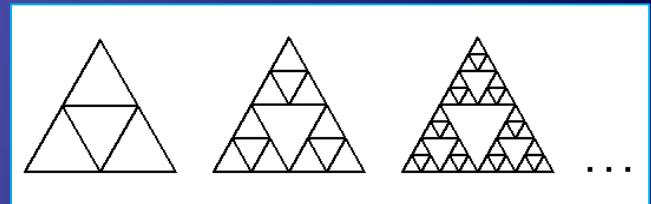
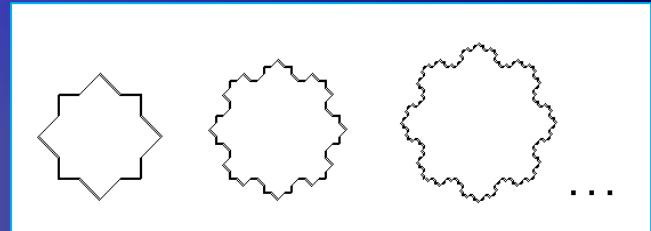
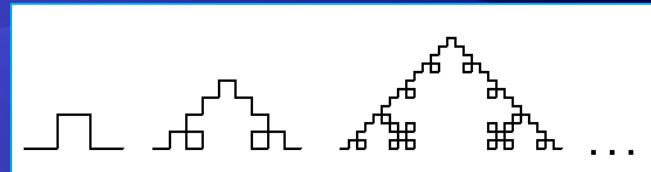
Cuvantul *fractal* a fost introdus de matematicianul Benoit Mandelbrot în 1975 și provine din latinescul *fractus*, care înseamnă spart sau fracturat.

Fractalul, ca formă geometrică, are următoarele caracteristici:

- este *auto-similar*: dacă se măreste orice porțiune dintr-un fractal, se vor obține cam aceleași detalii cu cele ale fractalului întreg,
- are o *definiție recursivă* – de exemplu:  $x, f(x), f(f(x)), f(f(f(x))), \dots$ ,
- are *detaliere și complexitate infinită*: orice nivel de magnificare pare identic și are o structură fină la scări infinit de mici.

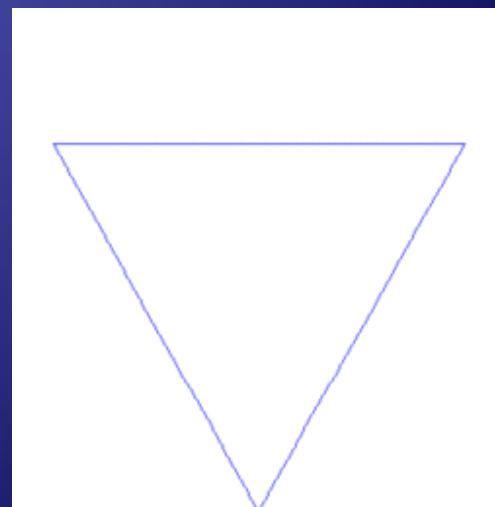
Elementele definitorii sunt:

- *initiator*: segmentul, curba sau forma initială.
- *generator*: regula folosită pentru a construi o nouă formă (*recurent/recursiv*).
- *iterație*: repetare / recursivitate



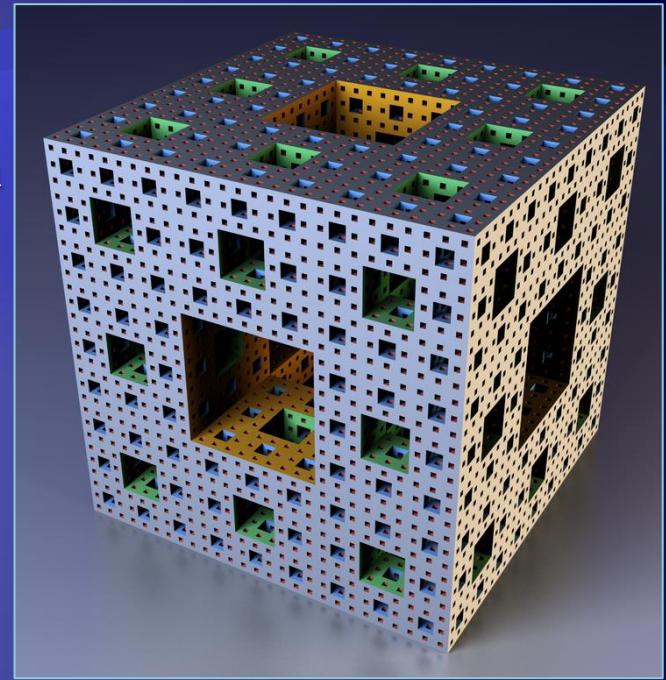
*Fractal*:

- Bază (rădăcină),
- Regula de dezvoltare (de creștere),
- Perioadă de creștere.



Fractali **renumiti**:

- *initiator*: segmentul, curba sau forma initială.
- *generator*: regula folosită pentru a construi o nouă formă (*recurent/recursiv*).
- *iteratie*: repetare / recursivitate







# Referinte

## Computer Graphics Abstracts

<http://www.ccs.neu.edu/home/fell/CS5310/abstracts/abstracts2011.html>

## Environment Maps Explained

[http://www.spiralgraphics.biz/genetica/help/index.htm?environment\\_maps\\_explained.htm](http://www.spiralgraphics.biz/genetica/help/index.htm?environment_maps_explained.htm)

## Hacking the Lite pt.1: Projective Texture Mapping

<http://www.derschmale.com/2010/12/30/hacking-the-lite-pt-1-projective-texture-mapping/>

## SIGGRAPH OpenGL Course Material

[Advanced Graphics Programming Techniques using OpenGL](#), SIGGRAPH 2000

[Advanced Graphics Programming Techniques using OpenGL](#), SIGGRAPH 1999

[Lighting and Shading Techniques for Interactive Applications](#), SIGGRAPH 1999

Advanced Graphics Programming Using OpenGL

<http://www.bluevoid.com/opengl/index.htm>

# *Temă*

Aplicați pe fețele unui corp (*poliedru*) cele trei *categorii de texturi*:

- a) *Constante,*
- b) *Variabile,*
- c) *Neregulate*

*Succes!*

