

# *Redarea suprafețelor 3D folosind texturi*

*Prof. univ. dr. ing. Florica Moldoveanu*

# Texturi (1)

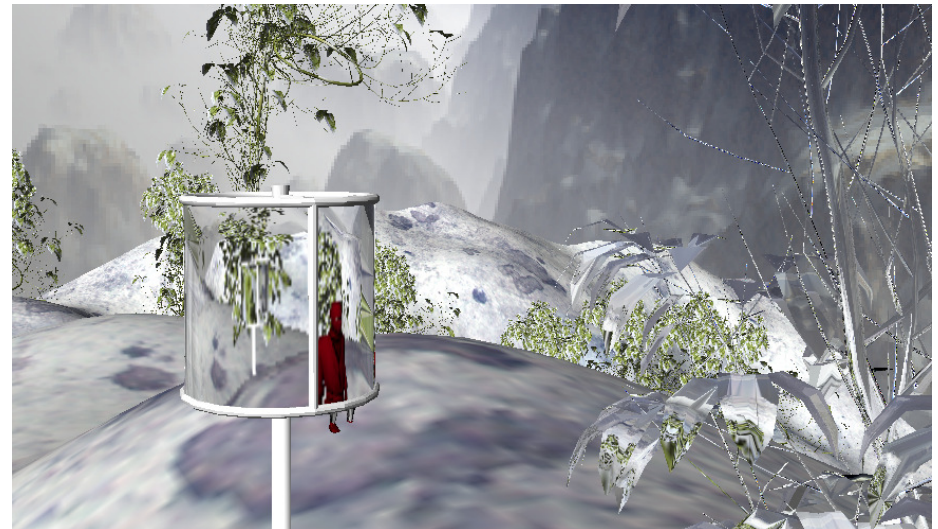
- Sunt doua tipuri de texturi:

1. Texturi reprezentate prin imagini sau șabloane care se aplica pe o suprafață neteda, suprafața rămânând netedă și după aplicarea texturii.

- Imaginea se aplica (« mapează ») pe suprafața în timpul generării imaginii suprafeței



- Imaginea mediului înconjurător redată pe suprafața unui obiect prin simularea reflexiei și a refracției.

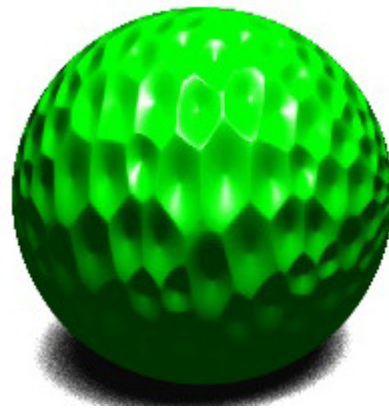


# Texturi (2)

2. Texturi prin care se redau detalii ale geometriei suprafetei.

Se folosesc diferite metode:

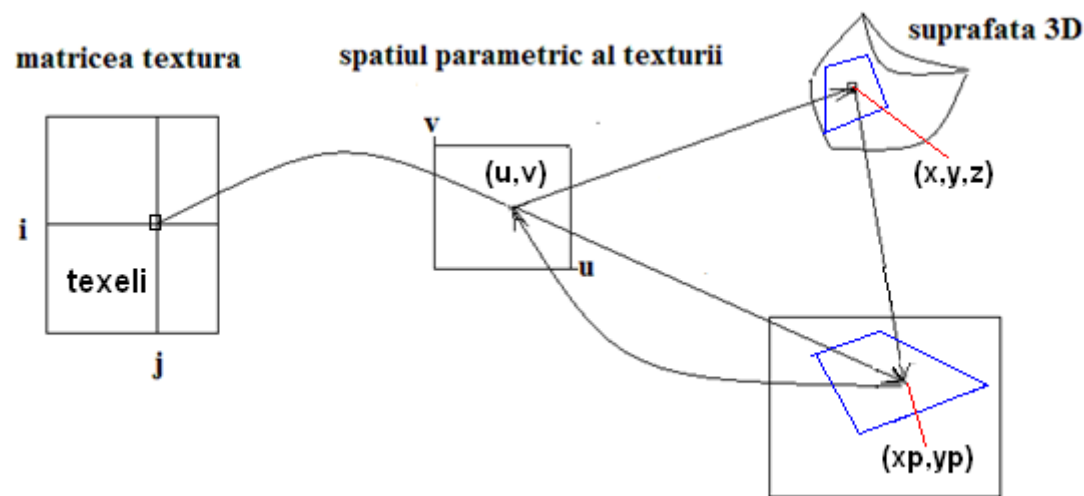
- perturbarea normalelor in punctele suprafetei
- metode fractale
- metode spectrale (functia textura este definita pentru componentele ei pe diferite frecvente)
- si altele



Redarea detaliilor geometrice prin perturbarea normalei la suprafata.

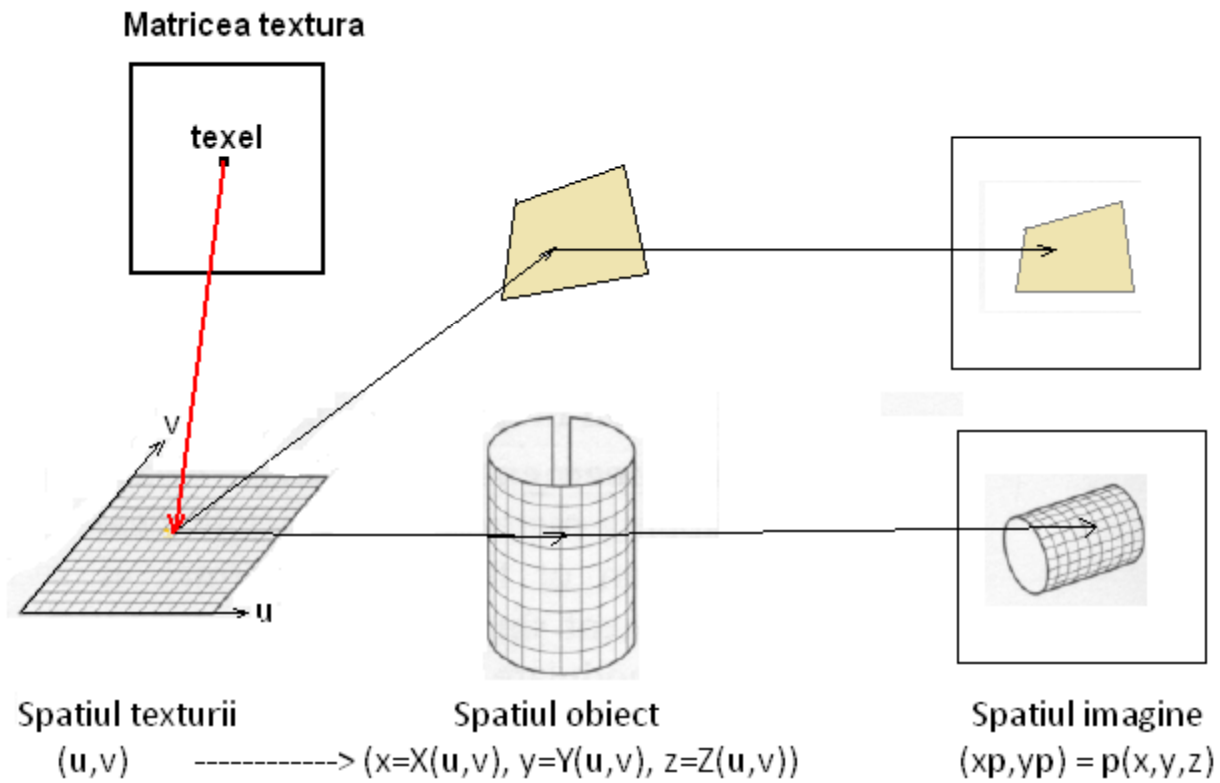
# Metode de aplicare a imaginilor textura

- Imaginea textura este memorata intr-o matrice ale carei elemente sunt culori si sunt adresate prin intermediul unui spatiu parametric  $(u,v)$ ,  $0 \leq u,v \leq 1$
- Elementele matricei textura sunt numite « texeli ».



- ❖ Sunt 2 metode de aplicare a imaginii textura pe imaginea unei suprafete 3D:
  1. “maparea directa” (forward mapping)
  2. “maparea inversa” (inverse mapping)

# Maparea directă(1)



## *Maparea directă(2)*

- Matricea textura este parcursă linie cu linie și pentru fiecare texel se calculează adresa pixelului în care se reprezintă.
- Imaginea de ieșire este completă numai după ce întreaga imagine textura a fost procesată. De aceea, este necesar un buffer în care să fie formată imaginea de ieșire.
- Metoda este adecvată atunci când imaginea textura se obține în timpul creerii imaginii de ieșire, linie cu linie.

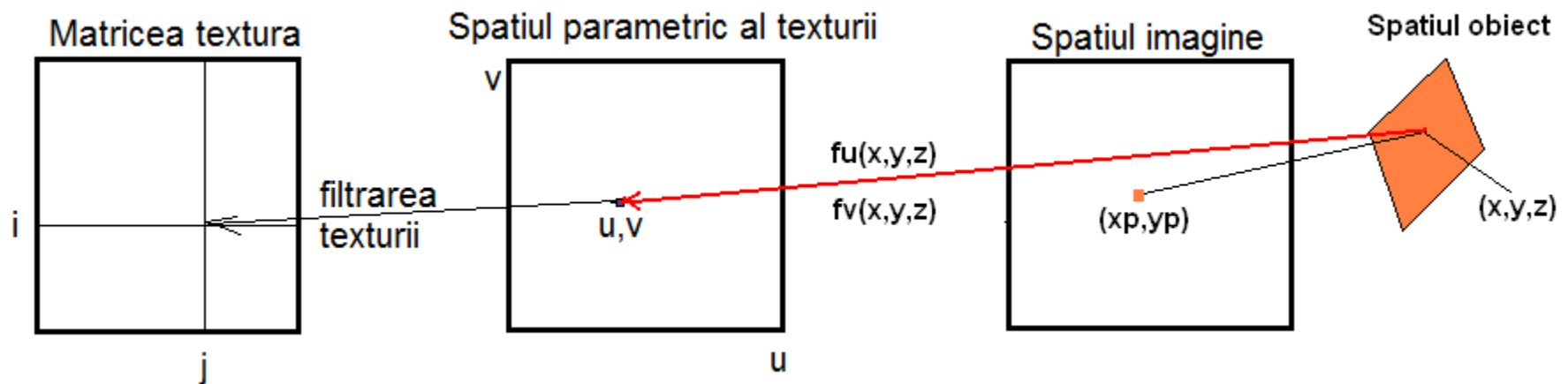
### **Dezavantaje**

- Pixelii sunt afișați într-o ordine arbitrară și nu se poate garanta că toți pixelii imaginii sunt asignați texturii.
- Dacă se consideră fiecare texel (pixel al imaginii de intrare) ca o suprafață, atunci el se mapează în imaginea de ieșire pe o suprafață de formă unui patrulater → sunt necesare calcule de intersecție dintre patrulater și celulele (pixelii) imaginii de ieșire.

# Maparea inversa(1)

- Este metoda uzuala de mapare a texturilor pe suprafetele 3D.
- Se pleaca de la spatiul imagine, aplicarea texturii fiind inglobata in procesul de rasterizare a primitivelor in care este descompusa suprafata 3D:
  - pentru fiecare pixel  $(x_p, y_p)$  in care se afiseaza un fragment al unei primitive se calculeaza adresa texelului/texelilor care trebuie folosit(i) in calculul culorii pixelului.

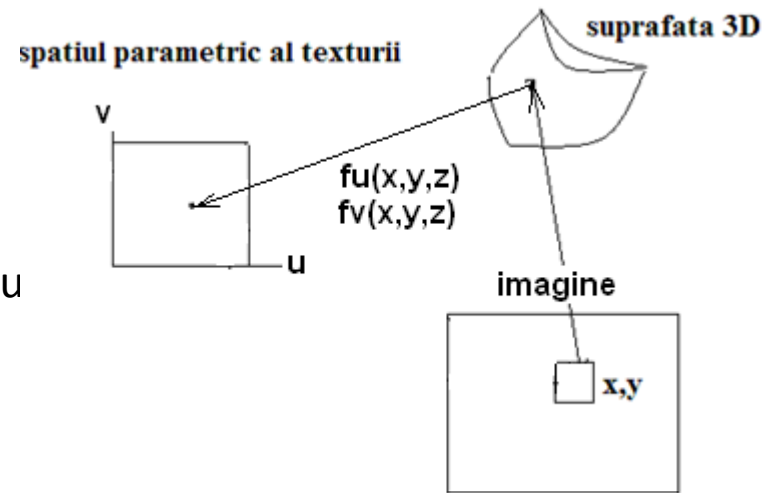
Adresa texelului se obtine folosind **functii de mapare inversa** si « **filtrarea texturii** »



# Maparea inversa(2)

Maparea inversa poate fi descompusa in doua etape :

1. Calculul punctului  $(x,y,z)$  de pe suprafata 3D  
corespunzator pixelului  $(x_p,y_p)$  in care se aplica textu
2. Calculul adresei  $(u,v)$  din spatiul texturii,  
corespunzatoare punctului de pe suprafata 3D.



La generarea imaginii se efectueaza o proiectie ortografica asupra scenei 3D,  
deci  $x = x_p$  ,  $y = y_p$  iar  $z$  se calculeaza in functie de tipul suprafetei;

Functiile  $f_u(x,y,z)$  si  $f_v(x,y,z)$  care definesc corespondenta dintre punctele suprafetei 3D si punctele spatiului  $(u,v)$  se numesc functii de mapare inversa:

$$u = f_u(x,y,z) ; v = f_v(x,y,z)$$

Exista functii de mapare inversa numai pentru anumite tipuri de suprafete; ele se numesc **functii de mapare standard**.



# *Functii de mapare standard(1)*

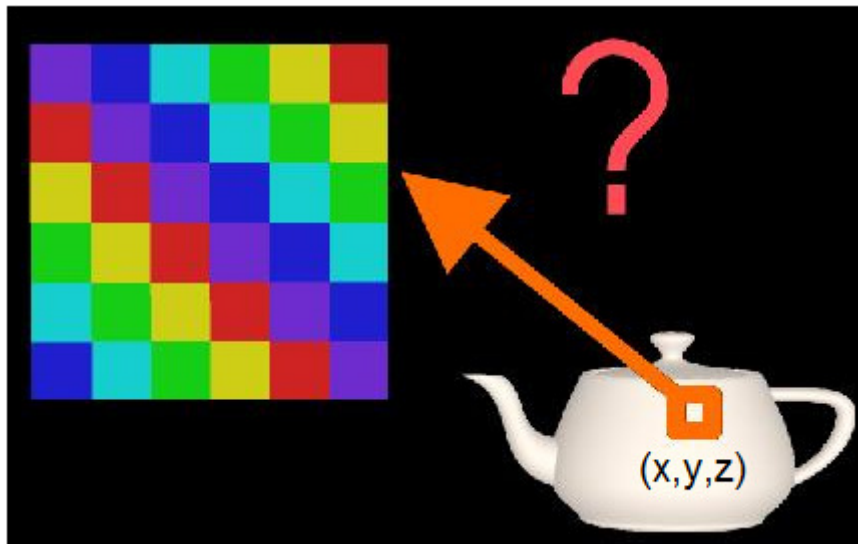
## 1. Maparea plana (maparea pe un plan)

Ecuatiile care descriu aceasta mapare sunt:

$$\begin{cases} u = s_u \cdot x - ou & x \in X \\ v = s_v \cdot y - ov & y \in Y \end{cases} \quad 0 \leq u, v \leq 1$$

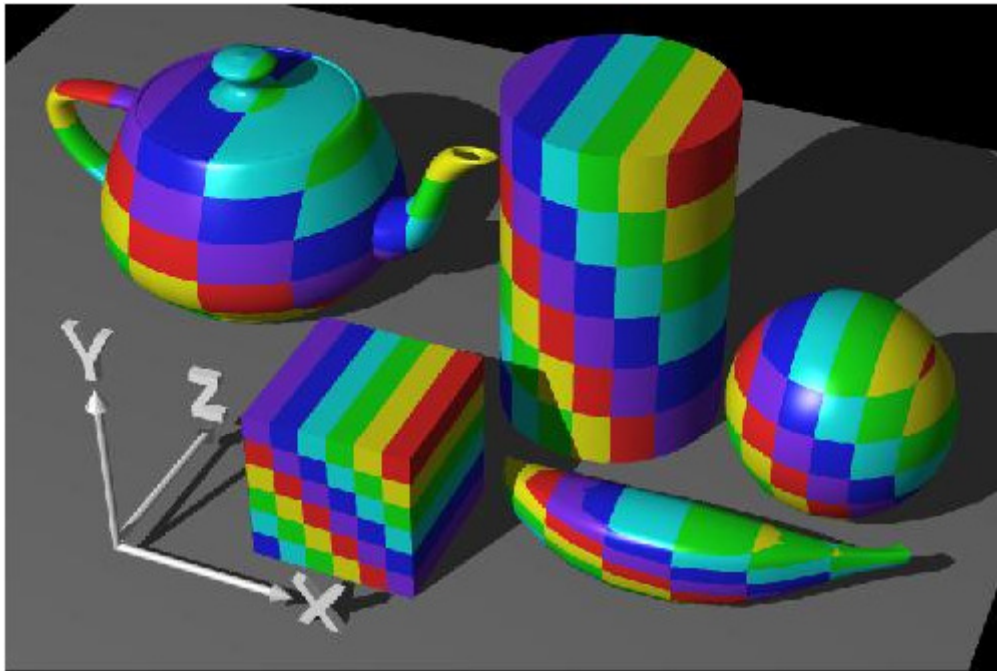
unde :  $s_u, s_v$  sunt factorii de scalare pentru transformarea spatiului  $(X, Y)$  in  $(u,v)$  iar  $(ou, ov)$  este originea texturii: determina modul in care se aseaza textura pe suprafata.

$(u, v) \leftarrow (x, y) \leftarrow (x, y, z)$  - se renunta la coordonata z!



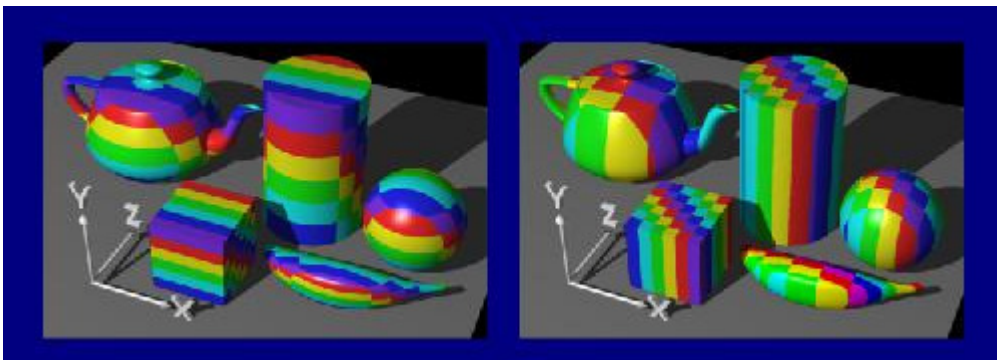
[http://www.siggraph.org/education/materials/HyperGraph/mapping/r\\_wolfe/r\\_wolfe\\_mapping\\_2.htm](http://www.siggraph.org/education/materials/HyperGraph/mapping/r_wolfe/r_wolfe_mapping_2.htm)

# Exemple de mapari plane



**Maparea plana prin renuntarea la coordonata z:** toate punctele suprafetei cu aceleasi coordonate  $(x,y)$  au aceeasi culoare.

[http://www.siggraph.org/education/materials/HyperGraph/mapping/r\\_wolfe/r\\_wolfe\\_mapping\\_2.htm](http://www.siggraph.org/education/materials/HyperGraph/mapping/r_wolfe/r_wolfe_mapping_2.htm)



Maparea plana prin renuntarea la coordonata x, respectiv y.

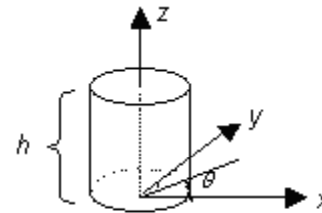
# Funcții de mapare standard(2)

## 2. Maparea cilindrica

Coordonatele (x,y,z) de pe suprafata sunt convertite in coordonate cilindrice, (θ, w).  
Ecuatiile parametrice ale unei suprafete cilindrice:

$$\begin{aligned} X &= r \cdot \cos(\theta) \\ Y &= r \cdot \sin(\theta) \\ Z &= w \cdot h \end{aligned}$$

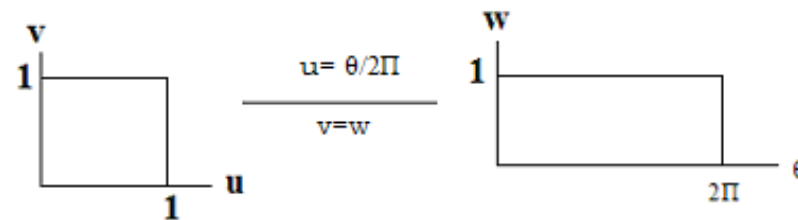
unde  $0 < \theta < 2\pi$ ,  $0 < w < 1$



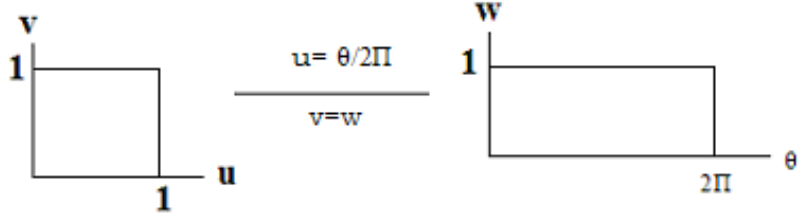
-Se trece din coordonate (x, y, z) in coordonate cilindrice

-Se pune in corespondenta spatiul parametric al texturii, (u,v), spatiului parametric al suprafetei:

$$\begin{cases} x = r \cdot \cos \theta \\ y = r \cdot \sin \theta \\ z = h \cdot w \end{cases} \Rightarrow \begin{cases} \theta = \arctg \frac{x}{y} \\ w = \frac{z}{h} \end{cases}$$



# Funcții de mapare standard(3)

$$\begin{cases} u = \frac{\theta}{2\pi} \\ v = w \end{cases}$$


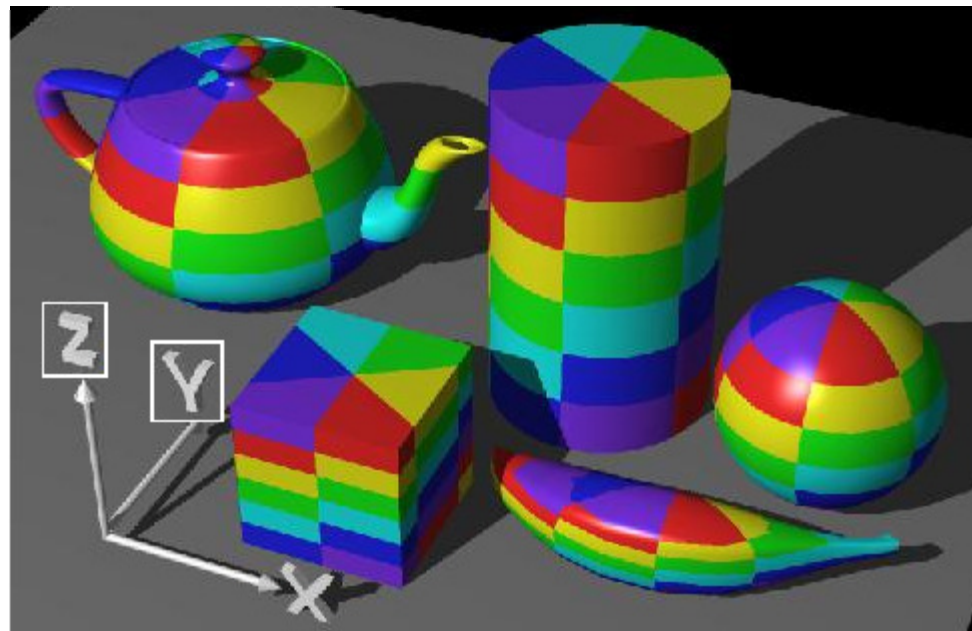
$$\begin{cases} u = \frac{1}{2\pi} \cdot \arctg\left(\frac{y}{x}\right) \\ v = \frac{z}{h} \end{cases}$$

## Funcția de mapare cilindrică

Textura este "înfasurată" în jurul obiectului.



Transformarea patratelor texturii în punctele suprafeței de  $z = z_{min}$  și  $z = z_{max}$

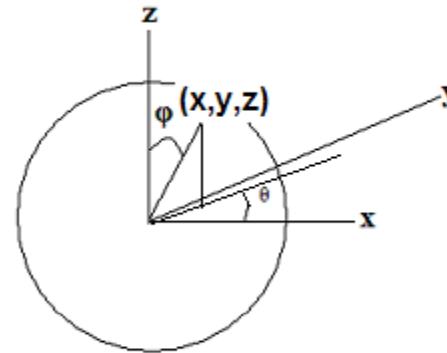


# Funcții de mapare standard(4)

## 3. Maparea sferica

- Se pleaca de la ecuatiile parametrice ale sferei
- Se trece din coordonate  $(x, y, z)$  in coordonate sferice

$$\begin{cases} x = r \cos \theta \sin \varphi \\ y = r \sin \theta \sin \varphi \\ z = r \cos \varphi \end{cases}, \text{cu } \begin{cases} 0 < \theta < 2\pi \\ 0 < \varphi < \pi \end{cases} \Rightarrow \begin{cases} \theta = \operatorname{arctg} \left( \frac{y}{x} \right) \\ \varphi = \arccos \left( \frac{z}{r} \right) \end{cases}$$



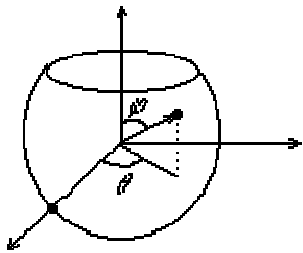
Rezulta

$$\begin{cases} u = \frac{\theta}{2\pi} \\ v = \frac{\varphi}{\pi} \end{cases} \text{ sau } \begin{cases} u = \frac{1}{2\pi} \operatorname{arctg} \left( \frac{y}{x} \right) \\ v = \frac{1}{\pi} \arccos \left( \frac{z}{r} \right) = \frac{1}{\pi} \arccos \frac{z}{\sqrt{x^2 + y^2 + z^2}} \end{cases}$$

**Funcția de mapare sferica**

# Funcții de mapare standard(5)

## Maparea pe un sector de sfera



$$\begin{cases} 0 \leq \theta \leq \frac{\pi}{2} \\ \frac{\pi}{4} \leq \phi \leq \frac{\pi}{2} \end{cases}$$

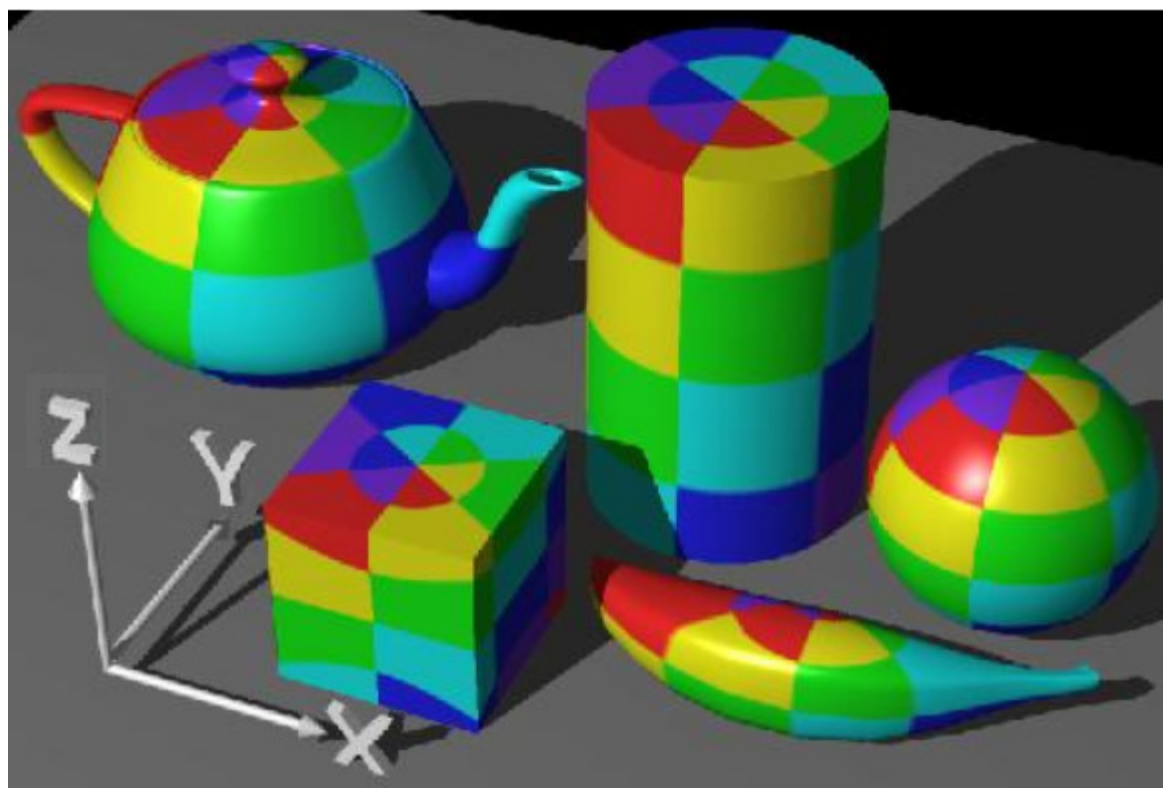
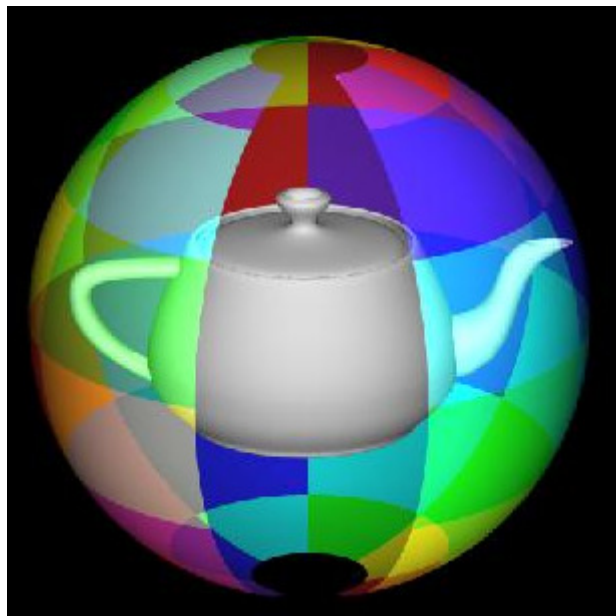
$$u = \theta / (\pi/2)$$

$$v = ((\pi/2) - \phi) / (\pi/4)$$

**Funcția de mapare**

# *Exemple de mapari sferice*

Maparea sferica "întinde" patratele texturii la nivelul ecuatorului si le "comprimă" proportional cu apropierea de poli.



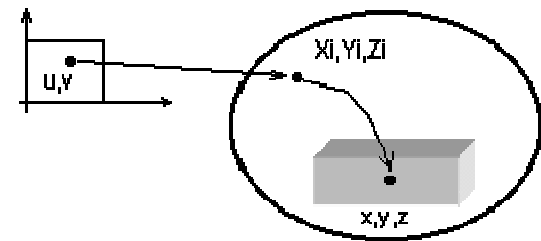
[http://www.siggraph.org/education/materials/HyperGraph/mapping/r\\_wolfe/r\\_wolfe\\_mapping\\_2.htm](http://www.siggraph.org/education/materials/HyperGraph/mapping/r_wolfe/r_wolfe_mapping_2.htm)

# Maparea în 2 pași(1)

- Este o tehnica de mapare independenta de forma obiectului pe care se aplica textura. Metoda a fost definita de Bier si Sloan (1986).
- Daca pentru suprafata texturata nu exista o functie de mapare inversa, atunci, se stabileste o corespondenta intre punctele suprafetei texturate si punctele unei suprafete 3D pentru care exista o functie de mapare inversa, numita in continuare « suprafata intermediara ».
- Aplicarea texturii este descompusa in doua operatii,

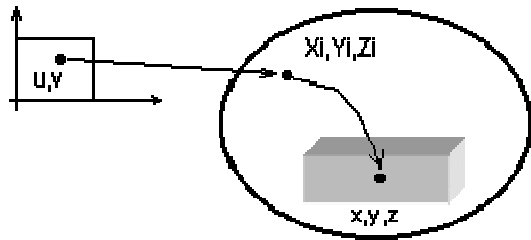
al caror efect este :

- **mularea texturii pe suprafata intermediara;**
- **mularea suprafetei intermediare pe suprafata de texturat.**





# Maparea în 2 pași(2)



a) **mapare S (mapare inversa):**

$$u = f_u(x_i, y_i, z_i)$$

$$v = f_v(x_i, y_i, z_i)$$

b)  $(x_i, y_i, z_i) \leftarrow$  **mapare O**  $\leftarrow (x, y, z)$

a) Mularea texturii pe suprafața intermediară se realizează prin funcția de mapare inversă. Aceasta este numită « **mapare S** », adică mapare pe *suprafață* intermediară.

b) Mularea suprafeței intermediare pe suprafața de texturat este numită « **mapare O** ». Ea trebuie să pună în corespondență fiecărui punct  $(x, y, z)$  al suprafeței (*obiectului*) de texturat, un punct  $(x_i, y_i, z_i)$  al suprafeței intermediare.

# *Maparea în 2 pași(3)*

## **Maparea S**

Se folosesc 4 tipuri de suprafețe intermediare :

1. un plan cu o orientare oarecare;
2. sfera;
3. cilindrul;
4. fețele unui cub.

Alegerea tipului suprafeței intermediare este dependentă de forma obiectului pe care trebuie să fie aplicată textura.

## **Maparea O**

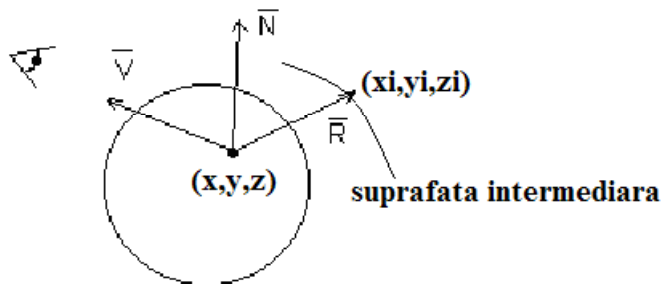
Autorii metodei au definit 4 tipuri de mapări O:

# Maparea în 2 pași(4)

## Mapari O

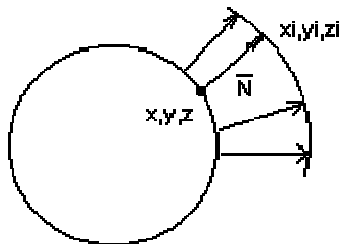
$(x,y,z)$  – un punct al suprafeței , pentru care se dorește calculul coordonatelor  $(u,v)$

$(x_i, y_i, z_i)$  – punctul corespunzător lui  $(x,y,z)$  pe suprafața intermediară



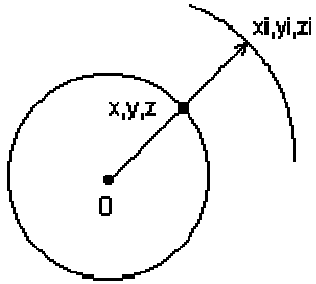
1.  $(x_i, y_i, z_i)$ , se obține intersectând suprafața intermediară cu vectorul reflectat al vederii în punctul  $(x,y,z)$ . R este simetric cu V față de N.

Metoda este dependentă de poziția observatorului (se folosește la « maparea mediului înconjurător »)

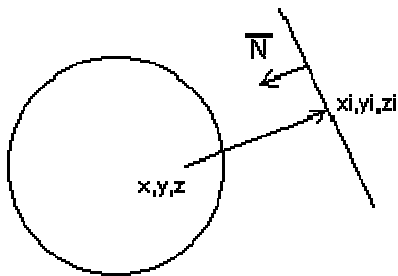


2.  $(x_i, y_i, z_i)$  este intersecția suprafeței intermediare cu normala la obiect în  $(x,y,z)$ .

## Maparea în 2 pași(5)



3.  $(x_i, y_i, z_i)$  este intersecția suprafeței intermediare cu dreapta care trece prin  $(x, y, z)$  și centroidul obiectului.



4.  $(x_i, y_i, z_i)$  este intersecția suprafeței intermediare cu dreapta care trece prin  $(x, y, z)$  și are direcția normalei la suprafața intermediară.

Acest tip de mapare O este corelat cu o mapare S pe un plan sau pe fețele unui cub.

Combinatia « mapare S pe cilindru » - « mapare O de tip 4 » este numita « shrinkwrap ».

# *Aplicarea texturilor pe fetele unei rețele poligonale 3D (1)*

In sistemele grafice actuale, orice suprafata 3D este descompusa, in vederea redarii sale, intr-o rețea de fațete poligonale/triunghiuri: textura se aplica pe poligoane la momentul rasterizarii lor.

Problema maparii inverse apare deci pentru fragmentele rezultate din rasterizarea poligoanelor.

Textura, definita in spatiul parametric pentru  $0 \leq u, v \leq 1$ , trebuie “intinsa” peste intreaga rețea de fațete poligonale.

Exista doua tipuri de **metode**:

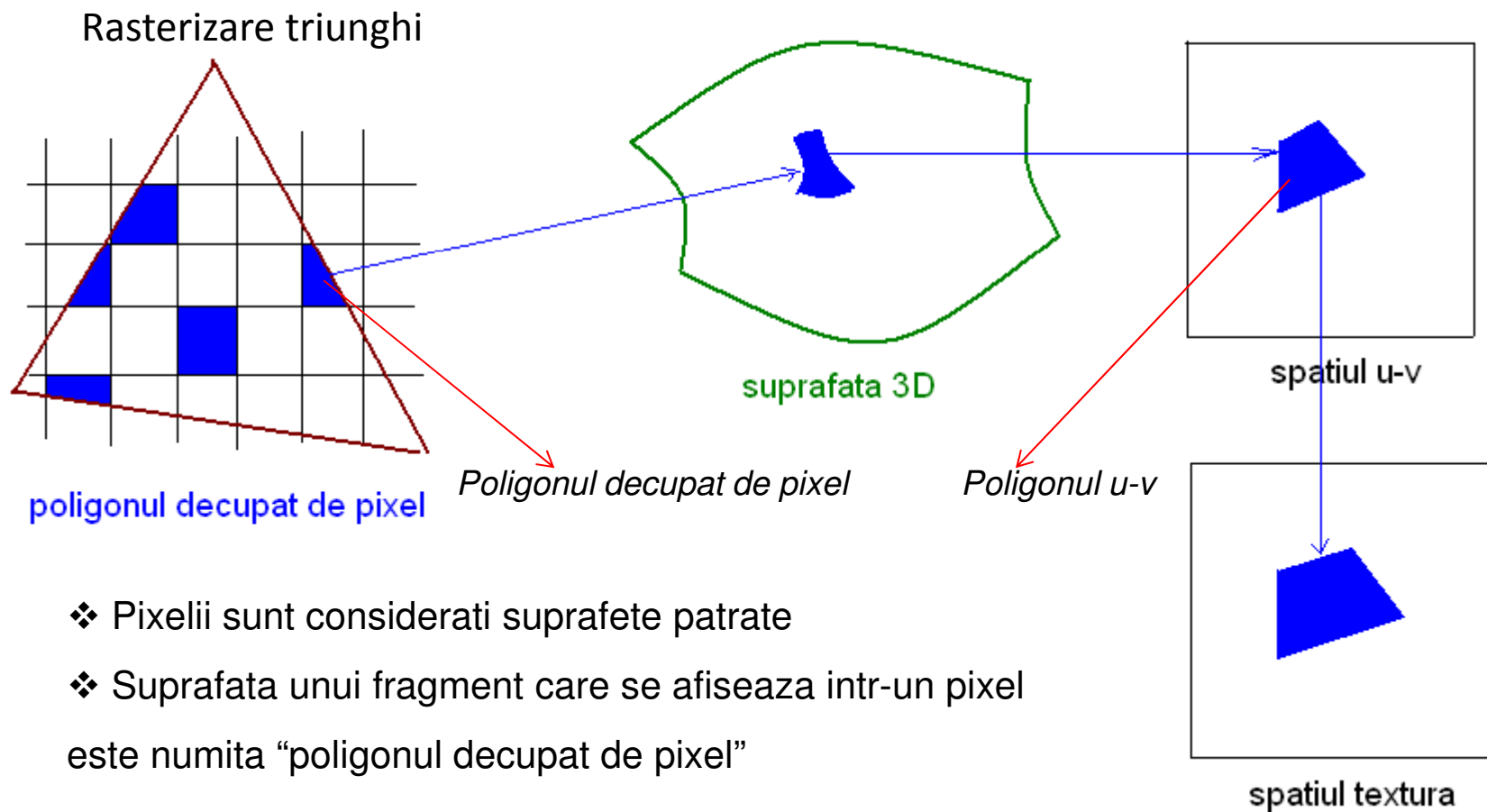
**Exacte**: se calculeaza  $(u, v)$  exact pentru fiecare fragment al unei fatete

**Aproximative**:

- se calculeaza  $(u,v)$  pentru fiecare varf al unei fatete
  - pentru o suprafata definita parametric  $S(u,v) = (x(u,v), y(u,v), z(u,v))$  se cunoaste  $(u,v)$  pentru fiecare punct de pe suprafata
  - pentru o suprafata oarecare,  $(u,v)$  se poate obtine printr-o *mapare in doi pasi*.
- se calculeaza  $(u, v)$  pentru fiecare fragment prin interpolare intre valorile  $(u,v)$  asociate varfurilor fatetei.

# Aplicarea texturilor pe fetele unei rețele poligonale 3D (2)

## Metode exacte(1)



- ❖ Pixelii sunt considerati suprafete patrute
- ❖ Suprafata unui fragment care se afiseaza intr-un pixel este numita “poligonul decupat de pixel”

# *Aplicarea texturilor pe fetele unei rețele poligonale 3D(3)*

## **Metode exacte (2)**

Algoritm:

**pentru fiecare fragment rezultat din rasterizarea unui poligon:**

- determina varfurile poligonului decupat de pixelul in care se afiseaza,  $(x_i, y_i)$ ;
- determina varfurile peticului de suprafata 3D corespunzator poligonului decupat de pixel,  $(x_i', y_i', z_i)$ , aplicand transformarea de vizualizare inversa;
- calculeaza varfurile  $(u_i, v_i)$  ale poligonului corespunzator in spatiul  $(u, v)$  – *poligonul  $u-v$* , utilizand o functie de mapare inversa sau o mapare in 2 pasi:  
$$(u_i, v_i) = (f_u(x_i', y_i', z_i), f_v(x_i', y_i', z_i))$$
- determina culoarea de afisare a pixelului folosind culorile texelilor corespunzatori *poligonului  $u-v$*  prin diferite metode.

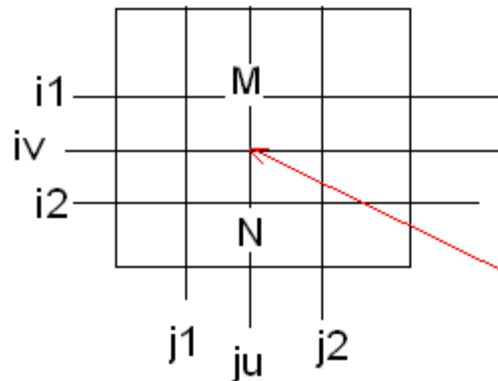
# Aplicarea texturilor pe fetele unei rețele poligonale $3D(4)$

## Metode exacte (3) - Calculul culorii pixelului : metoda 1

- Se calculează o valoare medie a coordonatelor varfurilor poligonului u-v :  
 $u_{med} = \text{suma}(u_i) / \text{nr. varfuri}; \quad v_{med} = \text{suma}(v_i) / \text{nr. varfuri};$
- Se calculează culoarea pixelului** folosind texelii din matricea textura, **prin interpolare biliniară.**

Dacă  $(j_u, i_v)$  sunt adresele reale corespunzătoare valorilor  $(u_{med}, v_{med})$  în spațiul textura,  $j_u = u_{med} / \text{nr\_col\_matrice\_textura}$ ,  $i_v = v_{med} / \text{nr\_linii\_matrice\_textura}$ , atunci:

Matricea textura



$$\text{Cul}(M) = \text{Cul}(i1, j1) + u_{med}(\text{Cul}(i1, j2) - \text{Cul}(i1, j1))$$

$$\text{Cul}(N) = \text{Cul}(i2, j1) + u_{med}(\text{Cul}(i2, j2) - \text{Cul}(i2, j1))$$

$$\text{Cul\_pixel} = \text{Cul}(M) + v_{med}(\text{Cul}(N) - \text{Cul}(M))$$

$i1, i2, j1, j2$  sunt indici de linie/coloana în matricea textura

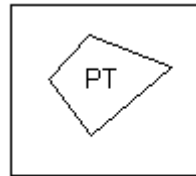
$j1 \leq j_u \leq j2, \quad i1 \leq i_v \leq i2$



# *Aplicarea texturilor pe fetele unei rețele poligonale 3D(5)*

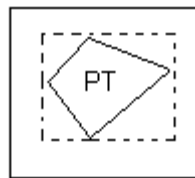
## **Metode exacte (4) - Calculul culorii pixelului: metoda 2**

- Se calculează poligonul corespunzător *poligonului u-v* în matricea texturii, PT



Matricea texturii

- Se aproximează poligonul texturii PT printr-un dreptunghi



Matricea texturii

- Se calculează media valorilor texelilor din dreptunghi.

# *Aplicarea texturilor pe fetele unei rețele poligonale*

## *3D(6)*

### **Metode aproximative**

- Se cunoaste  $(u,v)$  pentru fiecare varf al poligonului rasterizat.
- Pentru fiecare fragment rezultat din rasterizarea poligonului se calculeaza o adresa textura  $(u_f,v_f)$ , prin interpolarea adreselor textura,  $(u_i, v_i)$  asociate varfurilor (operatie integrata in algoritmul de rasterizare).
  - Pentru rezultate corecte este necesara efectuarea unei interpolari perspectiva. Interpolarea liniara este mai rapida dar nu produce efectul de perspectiva la aplicarea unei texturi pe suprafata unui poligon inclinat fata de planul de vizualizare.
- In cazul utilizarii mai multor texturi la redarea poligonului, fiecarui varf ii sunt asociate un acelasi numar de perechi de coordonate textura  $(u_i,v_i)$ ,  $1 \leq i \leq n$ , pentru accesarea celor  $n$  texturi;  $n$  este in general limitat la 8.
- Se calculeaza culoarea fragmentului folosind  $(u_f,v_f)$  sau  $(u_{fi}, v_{fi})$  printr-o metoda de "filtrare a texturii". Operatia este efectuata in "fragment shader".

# *Filtrarea texturii(1)*

- O textura este definita pe o latice discreta de puncte in timp ce coordonatele textura sunt valori reale. In general, o pereche (uf,vf) nu corespunde unui texel al texturii.
- **Metoda de calcul a culorii dintr-o textura, folosind coordonatele (uf,vf), se numeste “filtrarea texturii”.**

**Cazul  $0 \leq u_f, v_f \leq 1$**

**Sunt doua moduri standard de filtrare:**

1. Se considera culoarea texelului cel mai apropiat in spatiul laticei (se trunchiaza coordonatele): rapid dar poate produce defecte in imagine
2. Se efectueaza o interpolare biliniara intre culorile celor mai apropiati texeli in spatiul laticei:
  - scade viteza de calcul a culorii fragmentului
  - poate produce un efect de incetosare

# *Filtrarea texturii(2)*

## **Cazul $u_f, v_f$ in afara domeniului $[0,1]$**

- Coordonatele  $(u_i, v_i)$  asociate varfurilor pot fi in afara domeniului  $[0,1]$ . Rezulta:

coordonatele  $(u_f, v_f)$  ale unui fragment pot fi in afara domeniului  $[0,1]$ .

- Coordonatele  $(u_f, v_f)$  rezultate in procesul de rasterizare sunt transformate in coordonate

$$0 \leq u, v \leq 1$$

- Cele doua moduri standard de transformare a coordonatelor textura in domeniul  $[0,1]$  sunt denumite “clamping” si “wrapping (sau “repeating”): ele permit utilizarea eficienta a texturilor si obtinerea de efecte interesante.

## **Clamping (Comprimare)**

Coordonatele  $(u_f, v_f)$  sunt transformate in  $(u, v)$  astfel:

$$(u, v) = (\min(\max(0, u_f), 1), \min(\max(0, v_f), 1))$$

Deci: daca  $u_f$  sau  $v_f < 0$ , atunci  $u$ , respectiv  $v = 0$ ; daca  $u_f$  sau  $v_f > 1$ , atunci  $u$ , respectiv  $v = 1$

daca  $0 \leq u_f, v_f \leq 1$ , atunci  $u, v = u_f, v_f$ .

# *Filtrarea texturii(3)*

## **Clamping (Comprimare)**



**Aplicarea acestei transformari are ca efect propagarea culorii din imaginea textura de-a lungul directiei in care coordonata textura depaseste limita de 0 sau de 1.**

# Filtrarea texturii(4)

## Wrapping (Repetare)

- Pentru accesarea texturii se calculeaza (u,v) cu formula:

$$(u,v) = (uf - [uf], vf - [vf]),$$

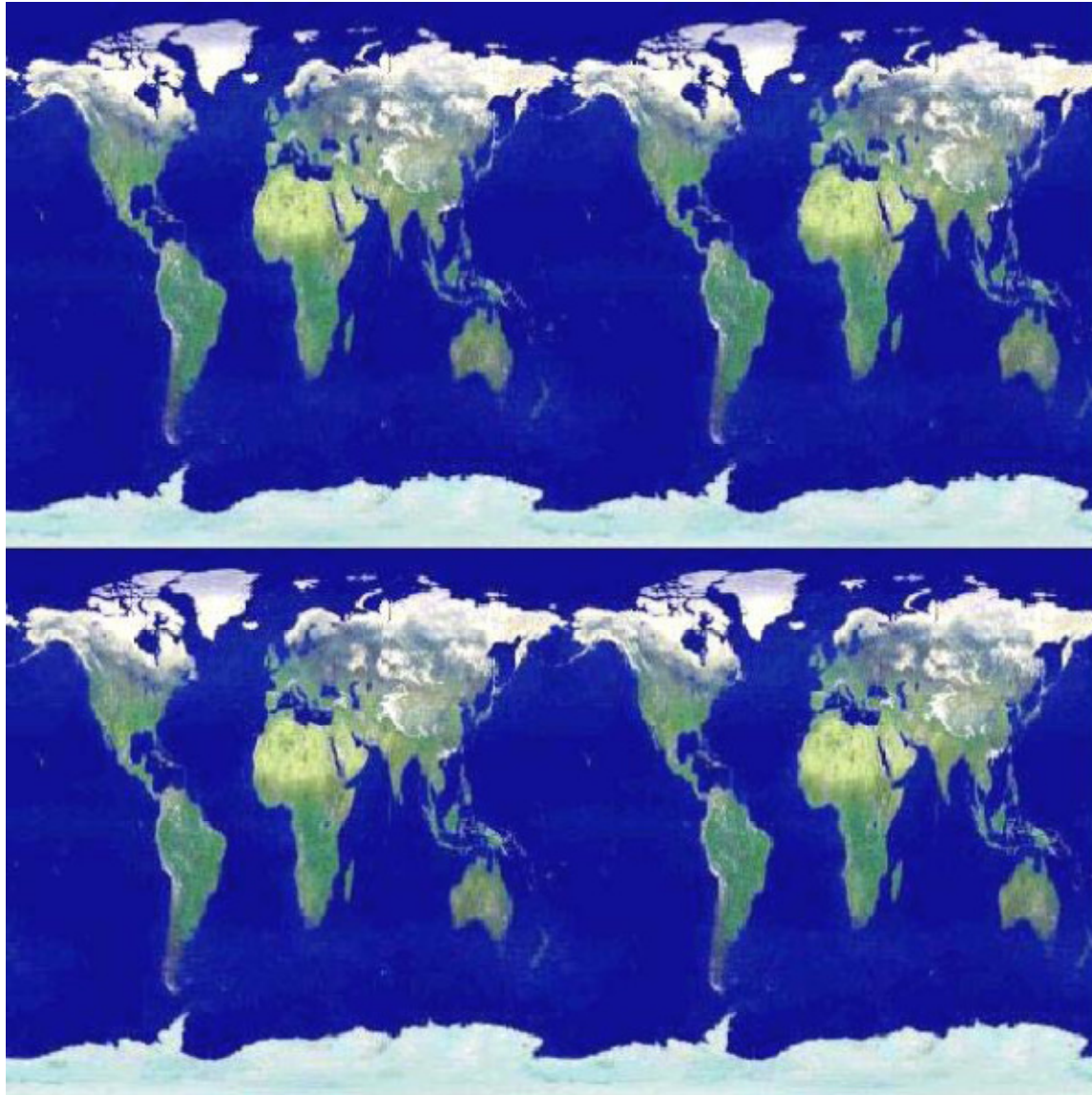
unde [w] reprezinta cel mai mare intreg mai mic sau egal cu w.

De exemplu, (-0.1, 1.5) se transforma in  $(-0.1 - (-1), 1.5 - (1)) = (0.9, 0.5)$ .

- **Acest mod de transformare permite repetarea unei texturi, producand un efect periodic.**
- La utilizarea acestui mod este necesar ca laturile stanga/dreapta, respectiv sus/jos ale imaginii textura se se potriveasca, astfel incat sa nu se observe marginile texturii.
- Daca pentru una dintre coordonate se utilizeaza transformarea “comprimare” iar pentru cealalta “repetare”, textura se numeste “cilindrica”.
- Daca pentru ambele coordonate se utilizeaza modul “repetare”, textura se numeste “toroidala”.

# *Filtrarea texturii(5)*

**Wrapping (Repetare)**



# *Transparența texturii*

- Imaginea textura poate avea o componentă adițională, numită “canalul alfa”, prin care se controlează transparența sau opacitatea texturii aplicate.
- Fiecare texel este în acest caz reprezentat prin 4 componente: (R,G,B,A), unde A reprezintă opacitatea.
  - dacă  $A = 1$ , atunci texelul este complet opac; dacă  $A=0$ , texelul este complet transparent.
- Pentru  $0 \leq A \leq 1$ , culoarea texelului modulează culoarea fragmentului, determinată pe baza geometriei (modelul Gouraud sau modelul Phong).
- Interfețele de programare OpenGL și Direct3D permit specificarea modului de combinare între culoarea unui fragment determinată pe baza geometriei primitivei și culoarea rezultată din filtrarea texturii (texturilor), folosind operatori care se aplică culorilor (RGB) și operatori care se aplică opacităților A.
- Combinarea dintre culoarea determinată pe baza geometriei și culoarea texturii poate fi programată în “fragment shader”.