

5ETI5 Synthèse d'images: Maillage

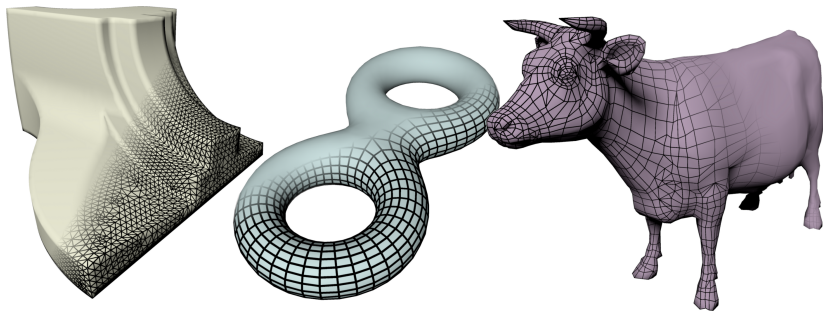
CPE Lyon
damien.rohmer@cpe.fr

2012/2013

- 1 Introduction Maillages polygonaux
- 2 Élement de base : Triangle
- 3 Description d'un maillage
- 4 Textures
- 5 Softwares

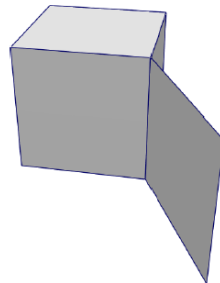
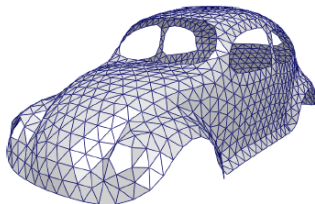
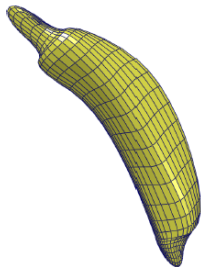
Maillage

- Maillage (Mesh) = Ensemble de **polygones** partageants certains sommets
- N_f **faces**, N_s **sommets** (vertices), N_e **arêtes** (edges).
- **Triangulation** : toutes les faces sont des triangles.
- **Quad-mesh** : toutes les faces sont des quads.
- Poly-mesh : mélange de types.



Topologie

- Rappel : Surface variété (Manifold) ssi le voisinage de tout point est homéomorphe à un (demi) disque.
⇒ Toute arête est partagée par au plus 2 faces (connectivité) + non auto-intersection (plongement).



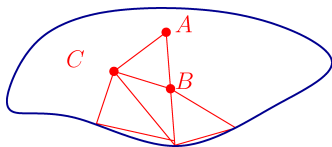
- Poly-mesh = cas particulier de triangulation
- Rappel : Triangulation = Mapping linéaire S

$$S_i : \begin{cases} \mathcal{D} \subset \mathbb{R}^2 & \rightarrow \mathbb{R}^3 \\ (u, v) & \mapsto S_i(u, v) = u\vec{AB} + v\vec{AC} \end{cases}$$

$$\mathcal{D} : 0 \leq u + v \leq 1$$

Propriétés :

- Surface globalement \mathcal{G}^0 .
- Surface jamais \mathcal{G}^1 (sauf plan).
- **Interpolation linéaire** de \mathbb{R}^2 vers \mathbb{R}^3 (normales, couleurs, textures).



Coordonnées dans un triangle

- Position du point p par rapport aux sommets (A,B,C) ?

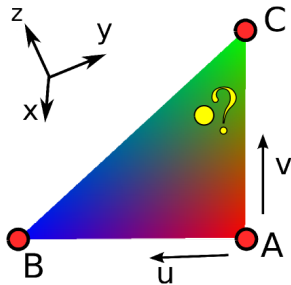
$$\vec{AP} = u\vec{AB} + v\vec{AC}$$

$$\Rightarrow P - A = u(B - A) + v(C - A)$$

$$\Rightarrow P = \underbrace{(1 - u - v)}_w A + uB + vC .$$

- (u,v,w) =Coordonnées barycentriques

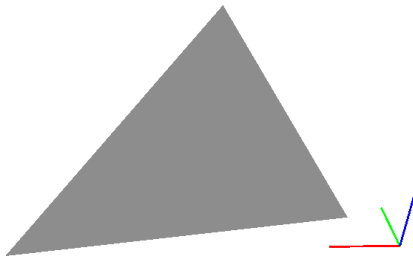
$$\begin{cases} P = wA + uB + vC \\ u + v + w = 1 \\ 0 \leq (u, v, w) \leq 1 . \end{cases}$$



Triangles en OpenGL

■ Syntaxe OpenGL (mode immédiat)

```
glBegin (GL_TRIANGLES) ;  
glNormal3d (0, 0, 1) ;  
glVertex3d (0, 0, 0) ;  
glVertex3d (1, 0, 0) ;  
glVertex3d (0, 1, 0) ;  
glEnd () ;
```



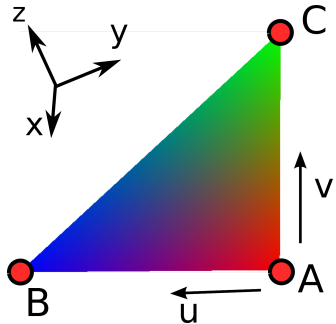
Interpolation linéaire

■ Interpolation de couleurs

$$\begin{cases} r(u, v) = (1 - u - v)r_A + ur_B + vr_C \\ g(u, v) = (1 - u - v)g_A + ug_B + vg_C \\ b(u, v) = (1 - u - v)b_A + ub_B + vb_C \end{cases}$$

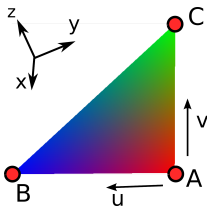
■ Dans le cas général pour une fonction f

$$f(u, v) = (1 - u - v)f_A + uf_B + vf_C$$



Interpolation de couleurs

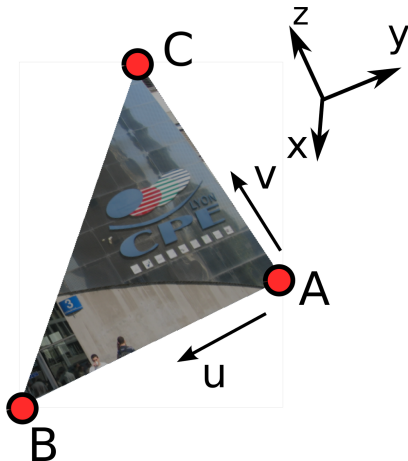
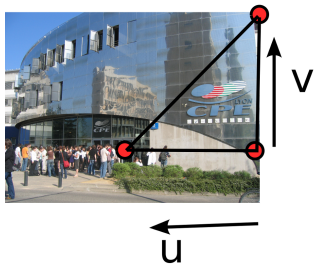
```
glBegin(GL_TRIANGLES);  
glNormal3d(0,0,1);  
glColor3d(1,0,0);  
glVertex3d(0,0,0);  
glColor3d(0,1,0);  
glVertex3d(1,0,0);  
glColor3d(0,0,1);  
glVertex3d(0,1,0);  
glEnd();
```



Interpolation linéaire

- On peut interpoler des coordonnées !
⇒ Textures

$$\begin{cases} t_x = (1 - u - v) t_x(A) + u t_x(B) + v t_x(C) \\ t_y = (1 - u - v) t_y(A) + u t_y(B) + v t_y(C) \end{cases}$$



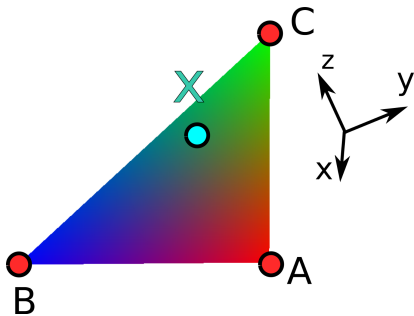
Coordonnées barycentriques

- Étant donné un point $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$: connaître (α, β, γ) tel que $\mathbf{x} = \alpha\mathbf{x}_A + \beta\mathbf{x}_B + \gamma\mathbf{x}_C$, $(\alpha + \beta + \gamma = 1)$.
⇒ coordonnées barycentriques.

$$\begin{cases} A = \text{aire}(\mathbf{x}_B - \mathbf{x}_A, \mathbf{x}_C - \mathbf{x}_A) \\ A_1 = \text{aire}(\mathbf{x}_C - \mathbf{x}_B, \mathbf{x} - \mathbf{x}_B) \\ A_2 = \text{aire}(\mathbf{x}_A - \mathbf{x}_C, \mathbf{x} - \mathbf{x}_C) \\ A_3 = \text{aire}(\mathbf{x}_B - \mathbf{x}_A, \mathbf{x} - \mathbf{x}_A) \end{cases}$$

avec $\text{aire}(\mathbf{v}_0, \mathbf{v}_1) = 1/2 \|\mathbf{v}_0 \times \mathbf{v}_1\|$

$$\Rightarrow \begin{cases} \alpha = A_1/A \\ \beta = A_2/A \\ \gamma = A_3/A \end{cases}$$

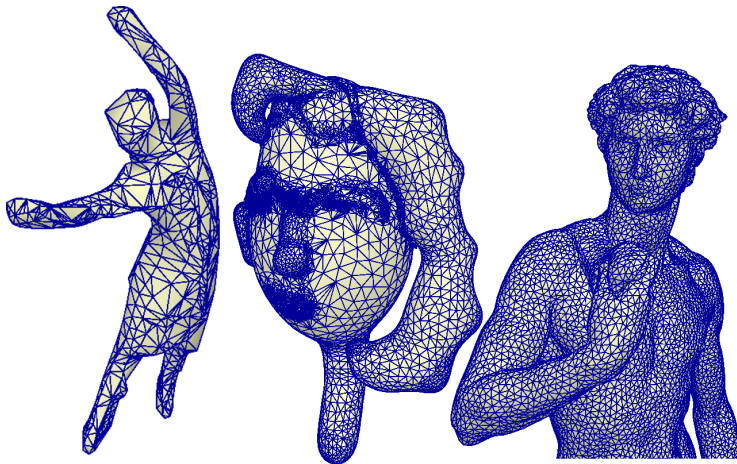


Qualité d'un maillage

- Triangulation : $\theta_{\min} \simeq 30^\circ$

- Quads : $\theta_{\min} \simeq 45^\circ$

Application : Calculs (FEM), (Rendu)



Maillage

Structure de données :

- *Ex. Représenter un tétraèdre :*

Idée 1 :

```
(0.0,0.0,0.0), (1.0,0.0,0.0), (0.0,0.0,1.0)
(0.0,0.0,0.0), (0.0,0.0,1.0), (0.0,1.0,0.0)
(0.0,0.0,0.0), (0.0,1.0,0.0), (1.0,0.0,0.0)
(0.0,1.0,0.0), (0.0,0.0,1.0), (1.0,0.0,0.0)
```

Il y a mieux :

coordonnees:

```
(0,0,0), (1,0,0), (0,1,0), (0,0,1)
```

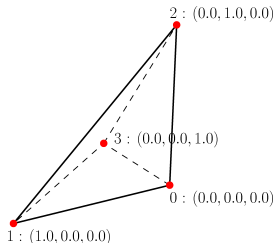
Connectivite

```
(0,1,3)
```

```
(0,3,2)
```

```
(0,2,1)
```

```
(1,2,3)
```

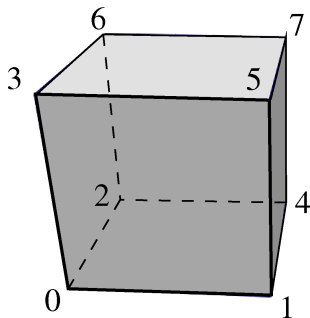


Format off

Exemple de format d'echange. Format off.

OFF

```
8 6 12
0 0 0
1 0 0
0 1 0
0 0 1
1 1 0
1 0 1
0 1 1
1 1 1
4 0 1 4 2
4 1 5 7 4
4 3 6 7 5
4 2 6 3 0
4 2 4 7 6
4 0 3 5 1
```



Lecture fichier

```
while(k_vertex<N_vertex)
{
    fscanf(fid,"%f %f %f",X,X+1,X+2);
    add_vertex(X[0],X[1],X[2]);
    k_vertex++;
}
for(k_poly=0;k_poly<N_poly;k_poly++)
{
    fscanf(fid,"%d",&size_poly);
    std::vector v_poly;
    for(k=0;k<size_poly;k++)
    {
        fscanf(fid,"%d",&temp);
        v_poly.push_back(temp);
    }
    add_polygon(v_poly);
}
```

Structure de données

- Vecteurs contigus dans la mémoire : Affichage rapide en OpenGL.

```
// (x0,y0,z0,x1,y1,z1,...)
std::vector <double> vertex

// (i00,i01,i02,i10,i11,i12,...)
std::vector <int> connectivity

std::vector <double> normal, color, texture ...
```

- Accès à la coordonnée y du sommet k .
`vertex[3*k+1]`
- Accès à la coordonnée y du sommet $s(1,2 \text{ ou } 3)$ du triangle t .
`vertex[3*connectivity[3*t+s]+1]`

Normale d'un maillage

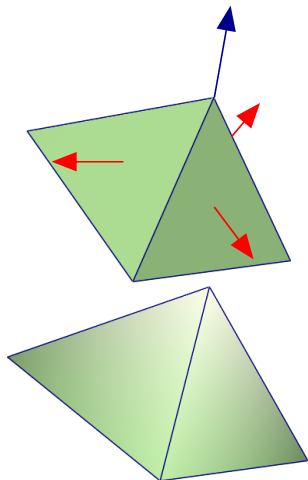
- Aspect lisse
⇒ 1 normale par sommet.
- Moyenne de normales (faux mais répandue)

$$\mathbf{n}_k = \frac{\sum_{i \in \mathcal{V}(k)} \mathbf{n}_i}{\left\| \sum_{i \in \mathcal{V}(k)} \mathbf{n}_i \right\|}$$

k : indice sommet

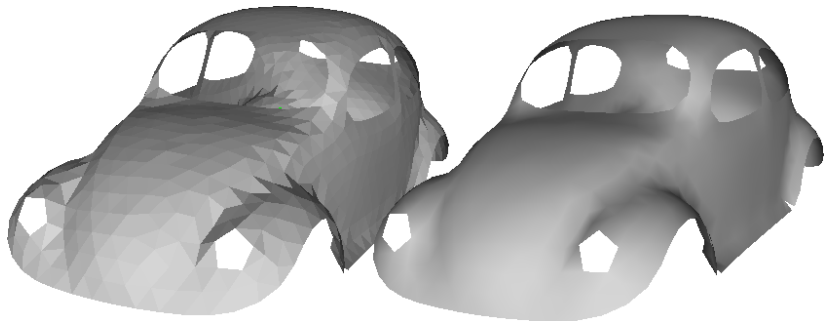
i : indice face

$\mathcal{V}(k)$: faces voisines du sommet k



Normales

- En OpenGL : 1 Normale interpolé par sommets
⇒ Normale par polygone = Plusieurs normales par sommet



Version lente

```
glBegin(GL_TRIANGLES);  
  for(k_tri=0;k_tri<N_tri;k_tri++)  
    for(k_vertex=0;k_vertex<3;k_vertex++)  
      for(k_dim=0;k_dim<3;k_dim++)  
        {  
          x[k_dim] = vertex[3*connectivity  
                                [3*k_tri+k_vertex]+k_dim];  
          n[k_dim] = normal[3*connectivity  
                                [3*k_tri+k_vertex]+k_dim];  
  
          glNormal3d(n[0],n[1],n[2]);  
          glVertex3d(x[0],x[1],x[2]);  
        }  
glEnd();
```

Version Rapide

```
glEnableClientState(GL_VERTEX_ARRAY);  
glVertexPointer(3, GL_DOUBLE, 0, &vertex[0]);  
  
glEnableClientState(GL_NORMAL_ARRAY);  
glNormalPointer(GL_DOUBLE, 0, &normal[0]);  
  
glDrawElements(GL_TRIANGLES, 3*N_tri,  
              GL_UNSIGNED_INT, &connectivity[0]);  
  
glDisableClientState(GL_VERTEX_ARRAY);  
glDisableClientState(GL_NORMAL_ARRAY);
```

Structure de données : Voisinage

- **1-Voisinage** = Sommets voisins d'un sommet donné

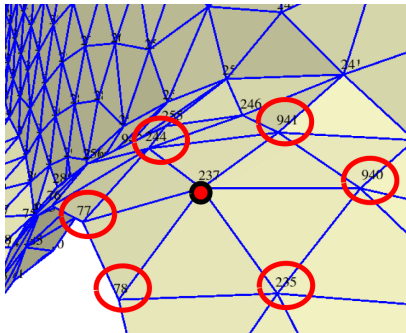
```
std::vector <std::vector <int> > one_ring
```

```
// exemple pour le cube:
```

```
one_ring[0] = [1,2,3]
```

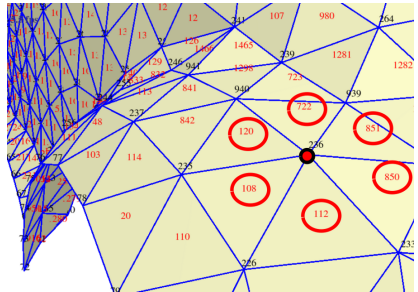
```
one_ring[1] = [5,4,0]
```

```
...
```



Voisinage

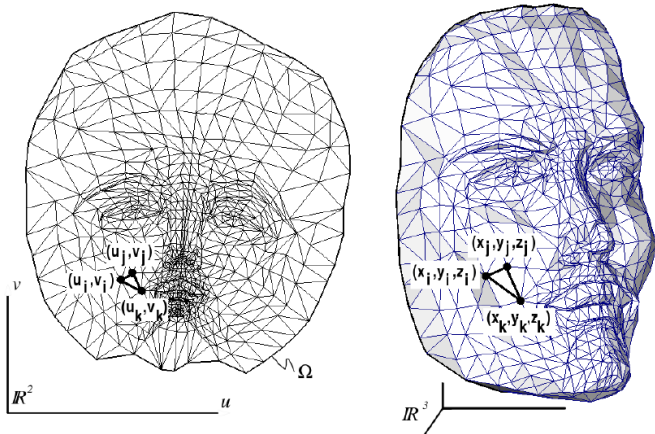
- Triangles voisins d'un autre
- Triangles voisins d'un point : étoilé (1-star)
⇒ calcul des normales !



! Attention aux structures de données.
Compromis entre : temps accès / temps recherche / espace
memoire / facilité ...

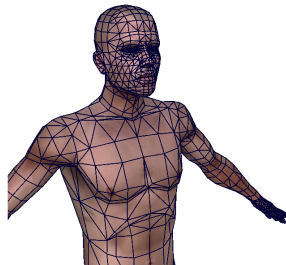
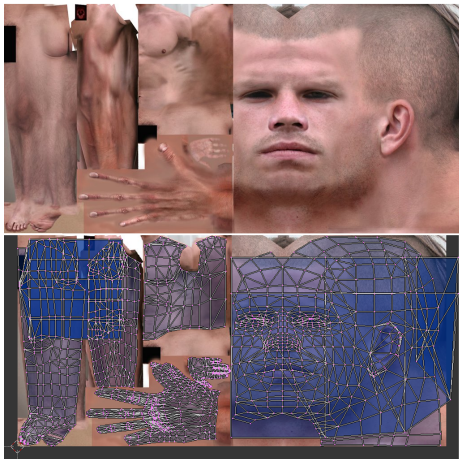
Paramétrisation / Textures

- Paramétrisation d'un maillage = Construction de S (par morceaux) étant donné Γ .

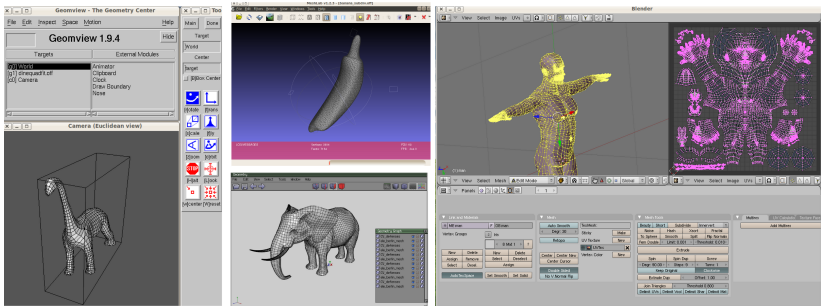


Textures

- Morceaux se recouvrants = atlas (charts).



Softwares



- Geomview (Viewer)
- Meshlab (Mesh Processing)
- Wings3D (Subdivision)
- Blender (Artiste)