# Plans, Polygones et Objects

Céline Loscos

CReSTIC

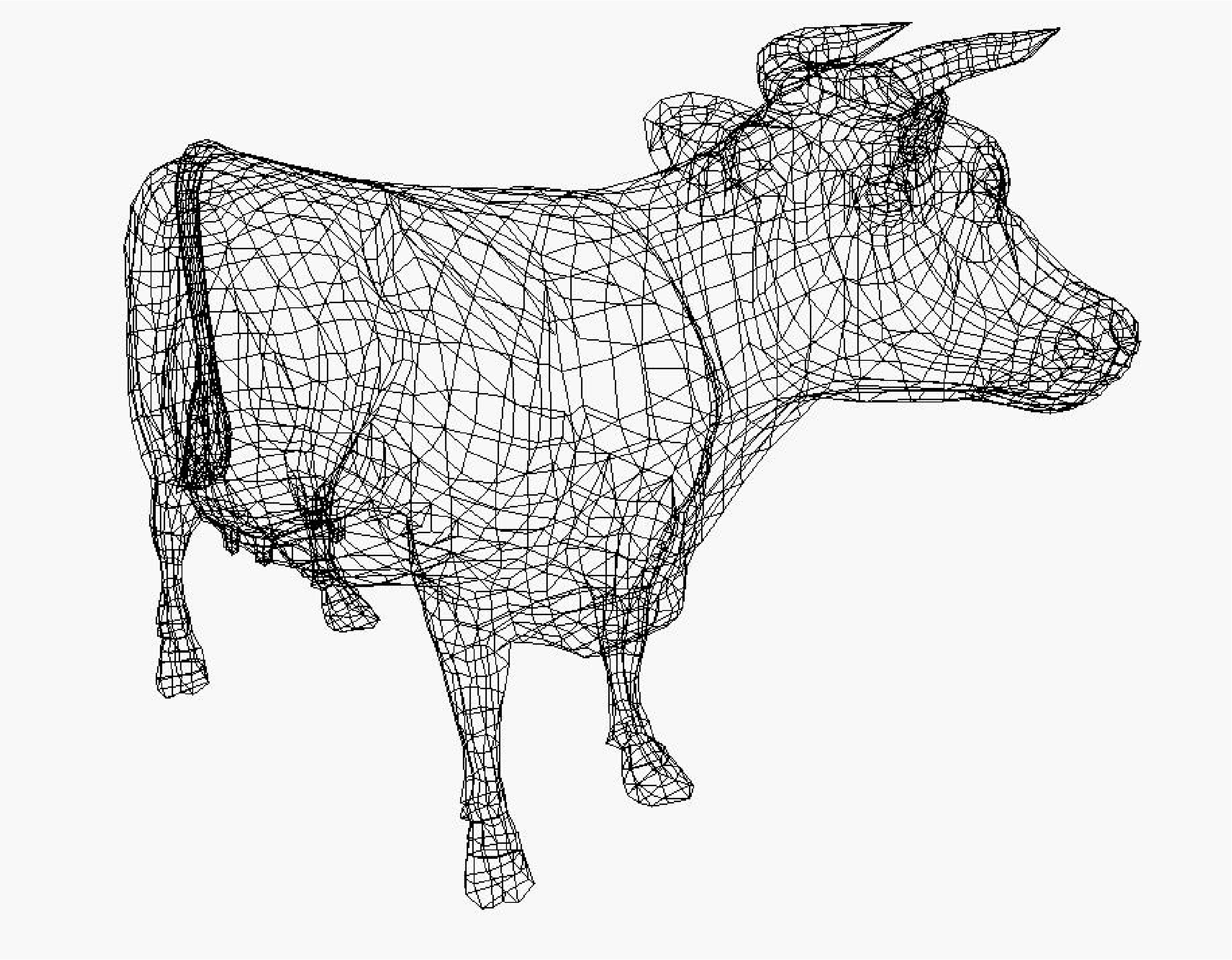
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

# Contenu

- Polygones
- Plans
- Créer des objets à partir de polygones

# Maillages de polygones

- La représentation en polygones est la plus fréquente



# Polygons

- Un polygone (face) Q est définie par une série de points

$$\left[p_0, p_1, p_2, \ldots, p_{n-1}, p_n\right]$$

$$p_i = \left(x_i, y_i, z_i\right)$$

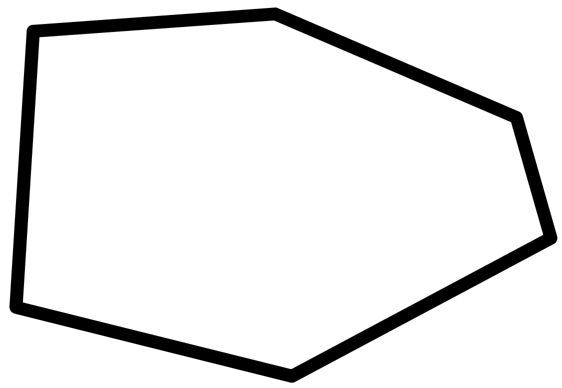
- Ces points doivent être **co-planaires**
- Trois points définissent un plan plane
  - Attention : un quatrième point ne sera pas obligatoirement sur ce plan !

CReSTIC

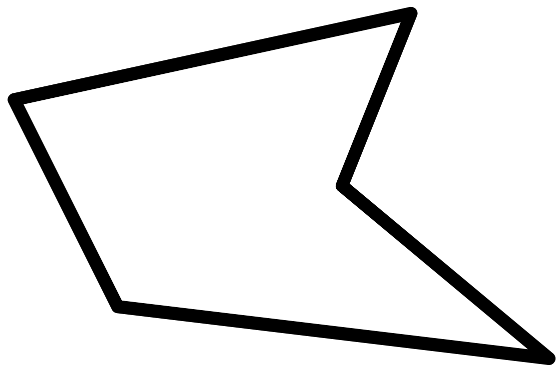
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

# Convexe, Concave

- Convexe



- Concave



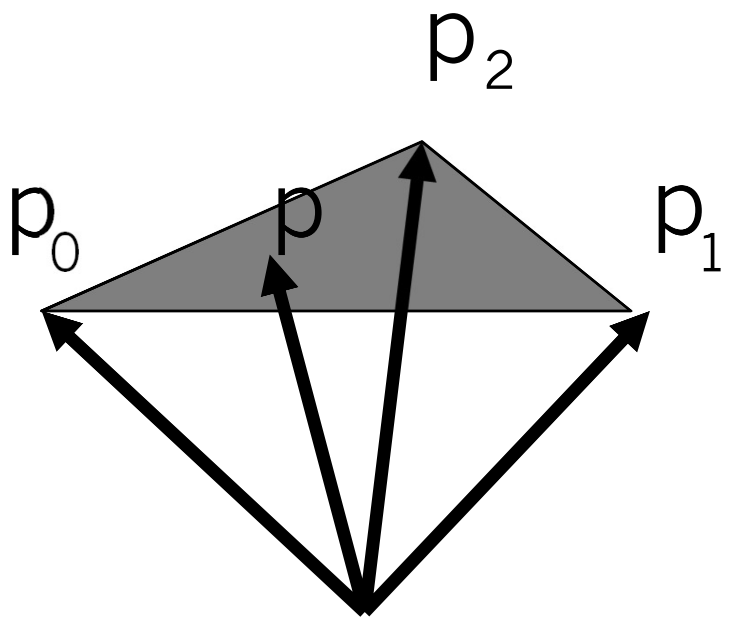
- Il est difficile de travailler avec des polygones concaves
- On va preferer les poplygones convexes, en particulier, les triangles !
- Il est toujours possible de découper les polygones en triangles

# Equation d'un plan

$$ax + by + cz = d$$

- a,b,c et d sont des constantes et définissent un plan unique
- x,y, z  forment un vecteur P.

# Retrouver a,b,c & d (1)




- Le produit vectoriel

$$n = (p_1 - p_0) \times (p_2 - p_0)$$

définit la **normale** du plan

- Un plan a deux normales (oppose)
- Les vecteurs dans le plan sont orhogonaux au vecteur normale du plan

# Deriving a,b,c & d (2)

- Comme $p$-$p_0$ est orthogonal à n

$$n \cdot \left(p - p_0\right) = 0$$

- Avec $n = (n_1,n_2,n_3)$
  - $a= n_1$ $b= n_2$ $c= n_3$ (n.p)
  - $d = n.p0 = n_1.x_0 + n_2.y_0 + n_3.z_0$

# Demi-espace

- Un plan coupe l'espace en deux **demi-espaces**
- Avec

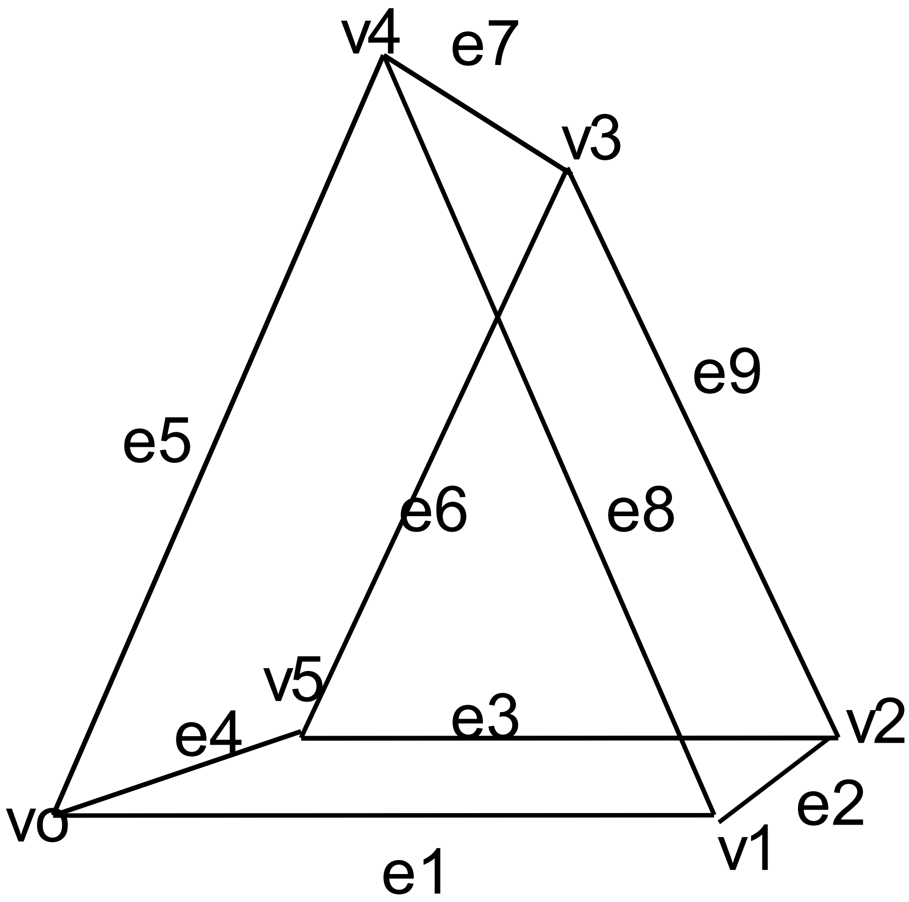
$$l(x, y, z) = ax + by + cz - d$$

- Si l(p) =0
  - Le point p est dans le plan
- Si l(p) > 0
  - Le point p est dans le demi-plan **positif**
- Si l(p) <0
  - Le point p est dans le demi-plan **négatif**

# Polyhèdre

- Les polygones sont assembles pour former des polyhèdres
  - Chaque coté (edge (E)) connecte deux sommets (vertex) et est la connexion entre deux polygones
  - Chaque sommet (vertex (V)) connecte 3 cotés
  - Aucunes faces (F) ne s'intersectent
- On a toujours la forumule suivante :
  - $V-E+F=2$

# Exemple de polyhèdre




- F0=v0v1v4
- F1=v5v3v2
- F2=v1v2v3v4
- F3=v0v4v3v5
- F4=v0v5v2v1

- V=6,F=5, E=9
- V-E+F=2

CReSTIC

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

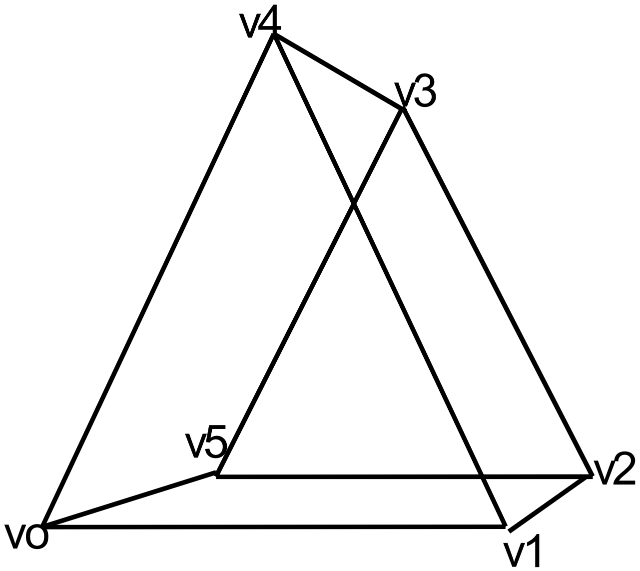
# Representation des Polyhèdres (1)

- Exhaustive (tableau de liste de sommets)
  - faces[1] = (x0,y0,z0),(x1,y1,z1),(x3,y3,z3)
  - faces[2] = (x2,y2,z2),(x0,y0,z0),(x3,y3,z3)
  - etc ....
- Stockage très inefficace car chaque sommet apparait au moins 3 fois !

# Representation des Polyhèdres (2)

- Ensemble des faces indexées
- Tableau de sommets array
  - vertices[0] = (x0,y0,z0)
  - vertices[1]=(x1,y1,z1)
  - etc ...
- Tableau de faces (liste des indices du tableau de sommets)
  - faces[0] = 0,2,1
  - faces[1]=2,3,1
  - etc ...

# L’ordre des sommets est important !

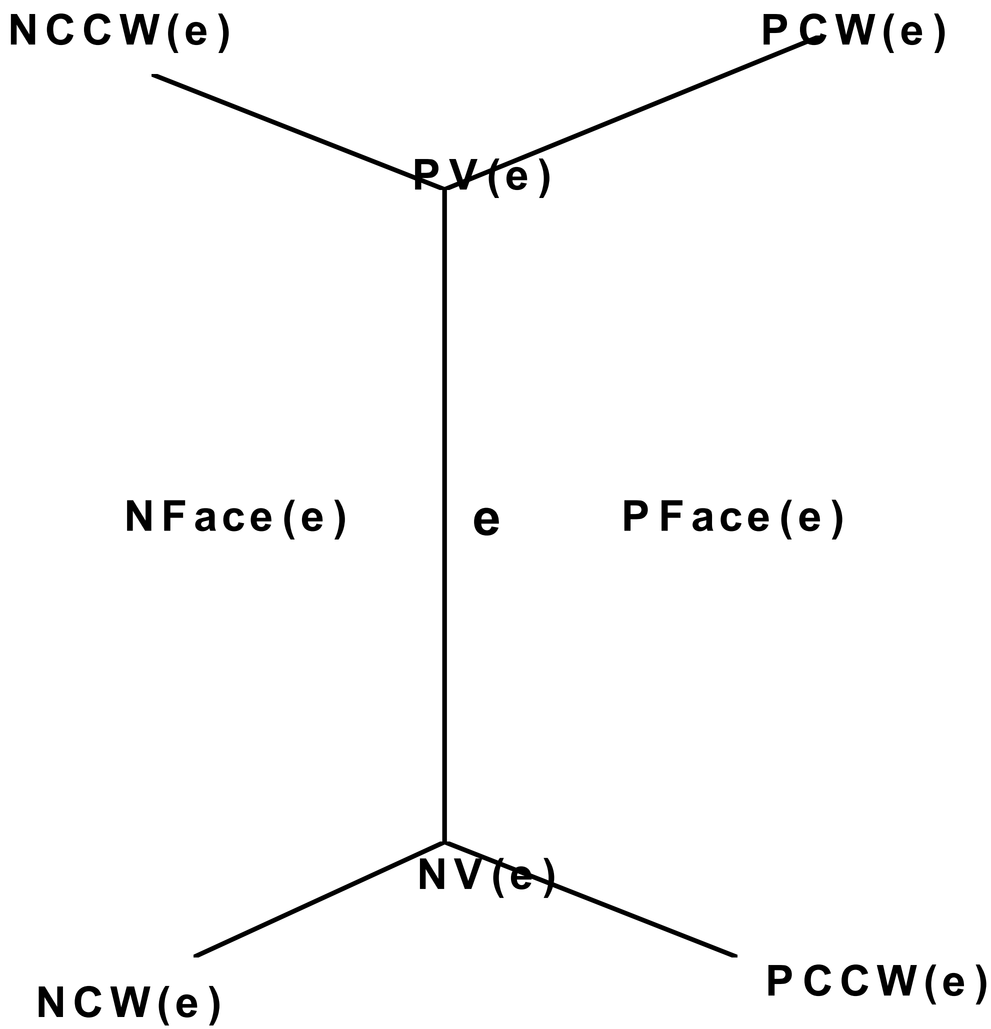



- Les polygones v0,v1,v4 et v0,v4,v1 ne sont PAS égaux
- La normale pointe dans une direction différente
- Habituellement, un polygone n’est affiché que s’il est visible depuis un point de vue situé dans le demi-espace positif
- C’est le principe du **back-face culling**

# Representation des Polyhèdres (3)

- Même une structure de faces indexées a une perte d'espace de stockage
  - Chaque coté est représenté deux fois
- Solution : Winged edge data structure (WEDS)
  - Liste de sommets
  - Liste de cotés (paires de sommets)
  - Liste de face (listes de cotés)

# Winged Edge Data Structure (WEDS)




- Edge e contains
  - Next edge NCW
  - Next edge NCCW
  - Prev edge PCW
  - Prev edge PCCW
  - Next face (NFace)
  - Prev face (PFace)
  - Next vertex (NV)
  - Prev vertex (PV)

# Advantages de la WEDS

- Une recherche simple est rapide
  - Pour trouver tous les cotés face
  - Pour trouver toutes faces associées à un sommet
  - etc…
- Operations complexes possibles
  - Couper un polygone en deux est facile (LOD)
  - Trouver une silhouette
  - Efficace pour le hardware
  - etc…

# Construire une WEDS

- Construire un ensemble de faces indexées
- Traverser chaque face dans l'ordre CCW pour construire les cotés
  - Nommer les sommets p et n, les faces p et n faces, et associer au coté CCW précédent
    - Compléter le prochain CCW du prochain coté de la face
    - Compléter le prochain CW et precedent CW en traversant la face adjacente

# Conclusion

- La modélisation en imagerie 3D s'appuie sur les concepts de géométrie
- Tous les objets sont décrits par un ensemble de polygones (concaves, quasiment toujours triangulaires)
- L'orientation des faces compte !
- Une structure de données permet :
  - D'avoir un ensemble cohérent, connecté et orienté des polygones pour former des polyhèdres
  - D'optimiser l'espace de stockage