

# GÉOMÉTRIE ALGORITHMIQUE

---

MODÉLISATION 3D

# INTRODUCTION

---

- La géométrie algorithmique telle qu'elle sera abordée dans ce cours concerne la représentation d'objets volumiques sur un écran (2D ou 3D).
- Plusieurs outils sont nécessaires pour obtenir une telle représentation:
  - Une structure d'objet simple permettant des calculs performants, polymesh composé de triangles
  - Des outils mathématiques pour adapter la structure 3D à l'écran, projection, rotation, redimensionnement
  - Des méthodes de rendu de l'intérieur des triangles pour représenter les lumières et les textures

# APERÇU DU PROGRAMME

---

- La modélisation 3D : 2h
  - TA:Théière Utah
- Les outils mathématiques: 6h
  - Trigonométrie
  - Vecteurs
  - Matrices
  - Quaternions
- La projection 3D -> 2D
  - Types de projection, caméras
  - Matrice écran, camera, monde
- TA: construction d'un moteur de projection 3D (fil de fer)



- Le rendu 3D (rasterization): 8h
  - Un peu de physique
  - Gouraud, phong, PBR
  - Mode de rendu avec shader
- TA: implémentation PBR (physical based rendering)
- Algorithmes avancés 3D: 8h
  - Ray tracing, réflexion, réfraction, optique
  - Triangulation, lissage de normales, ....
  - NURBS, courbes de Bézier
- TA: ray tracing logiciel



# MODÉLISATION 3D

---

CSG, B-REP, SPLINES, SURFACES IMPLICITES, VOXELS



# GÉOMÉTRIE CONSTRUCTIVE DES SOLIDES (CONSTRUCTIVE SOLID GEOMETRY CSG)

---

- Union et intersection de formes géométriques dans l'espace

- Description des objets sous formes d'équations

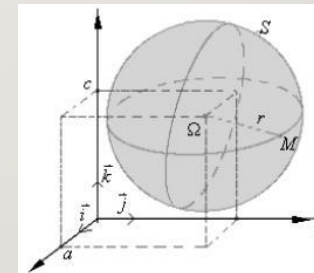
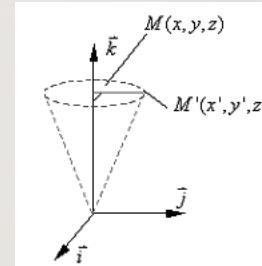
- Plan :  $ax + by + cz + d = 0$

- Cône :  $z^2 \cdot \tan^2 \theta = x'^2 + y'^2$

- Sphère :  $(x - a)^2 + (y - b)^2 + (z - c)^2 = r^2$

- Description de contours 2D purs extrapolés en 3D

- Représentation de tels objets nécessite de résoudre l'équation, pour chaque point de l'espace représenté par l'écran, temps de calculs très importants, possible avec le ray tracing mais encore lent et la représentation d'objets complexes nécessite de multiplier les équations. Il ne s'agit pas d'une représentation adapté à l'affichage temps réel.

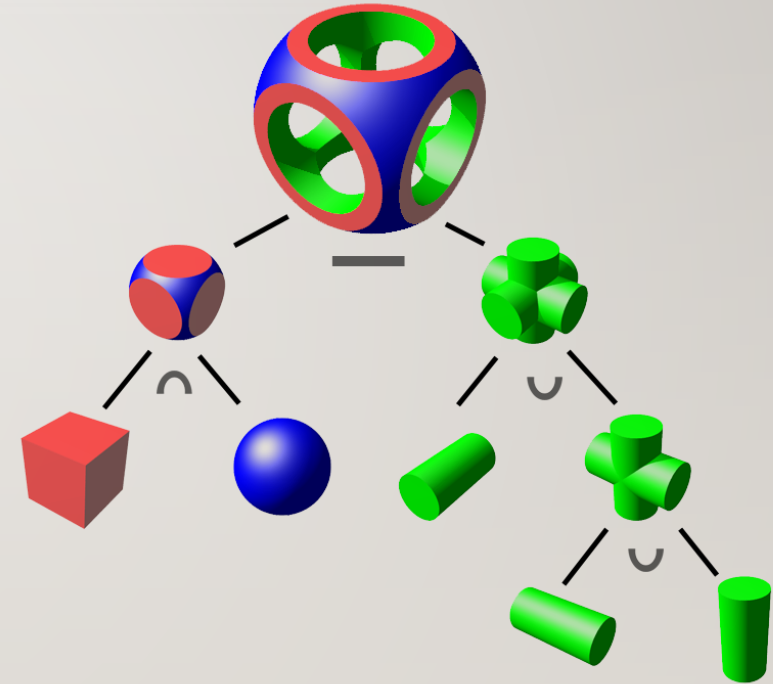




# OPÉRATION BOOLÉENNES

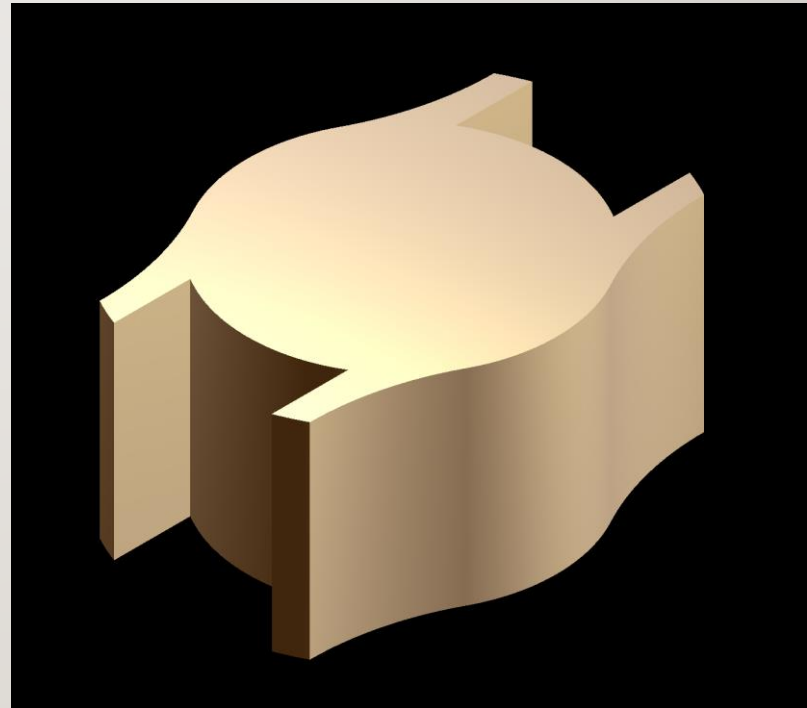
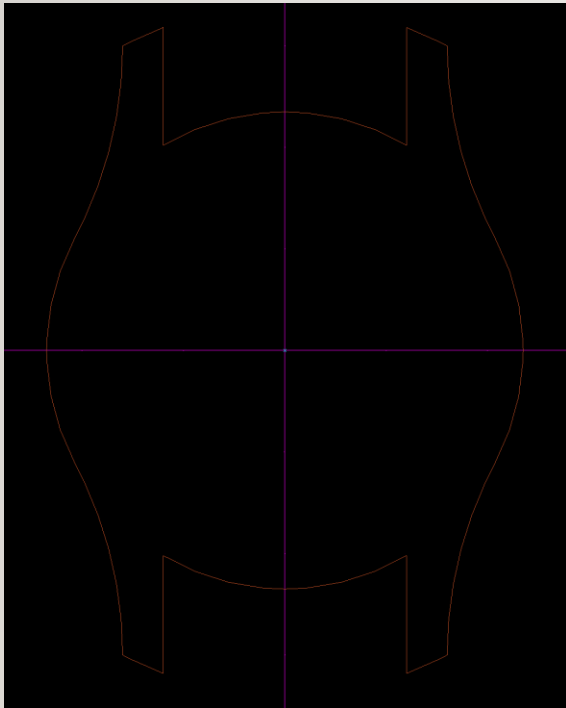
---

- Pour créer des formes complexes en CSG il est nécessaire de composer des formes simples avec des opérations telles que :
  - Union – satisfait cette équation OU cette équation
  - Intersection – satisfait cette équation ET cette équation
  - Soustraction- satisfait cette équation MAIS PAS cette équation



# ELEVATION

---

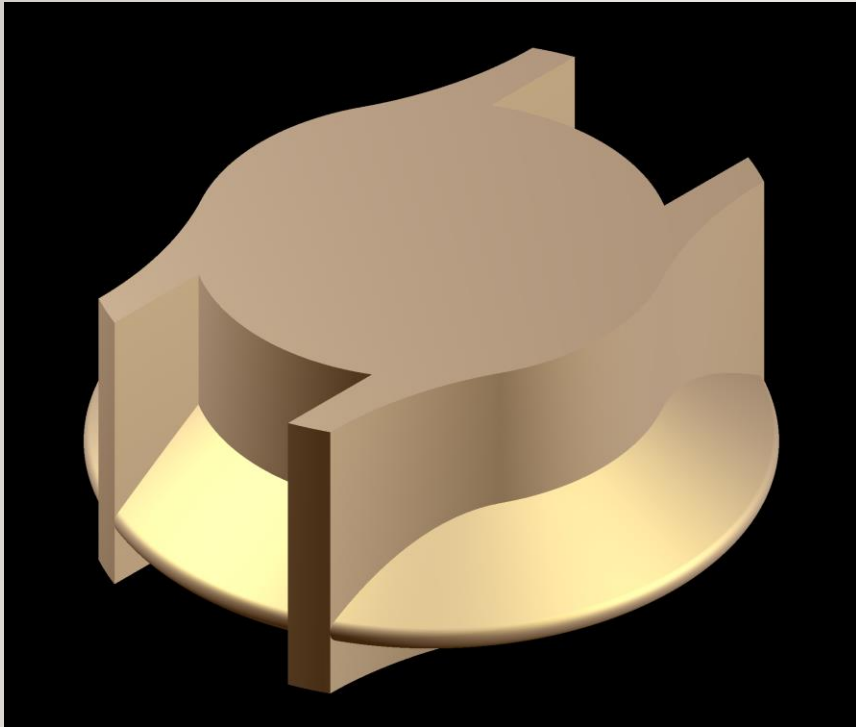






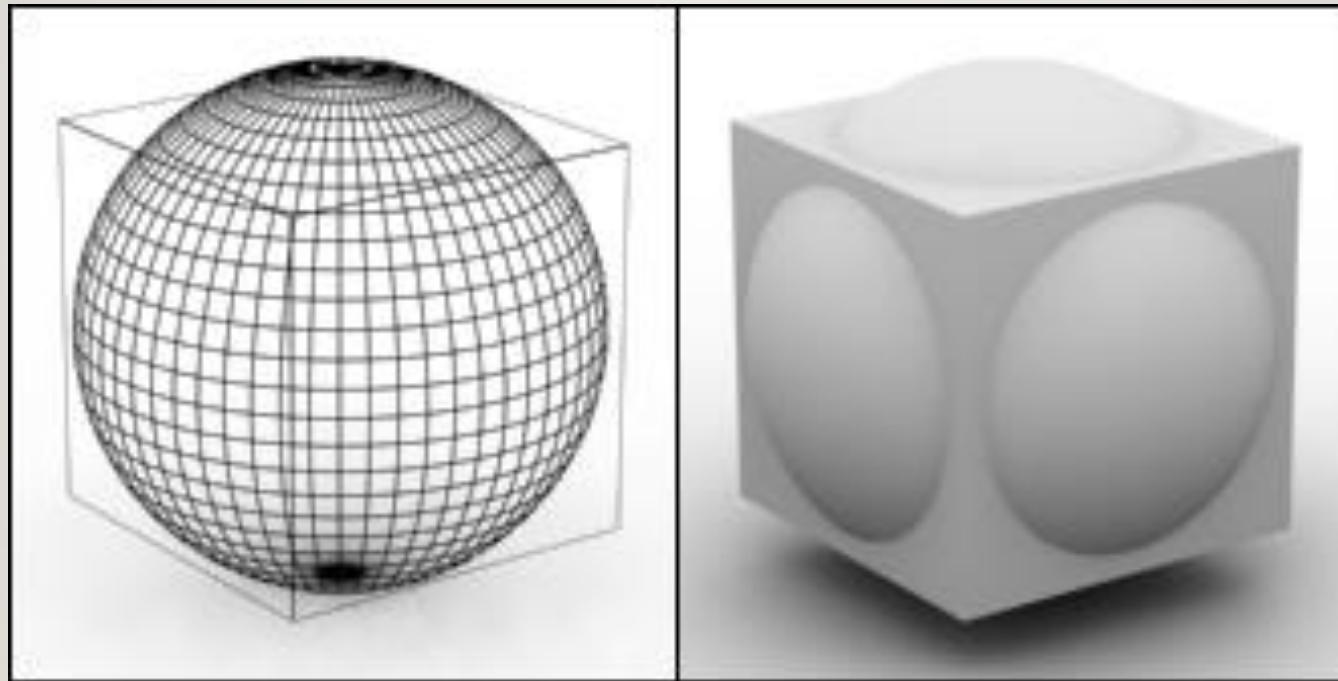
# INTERSECTION

---



# UNION

---



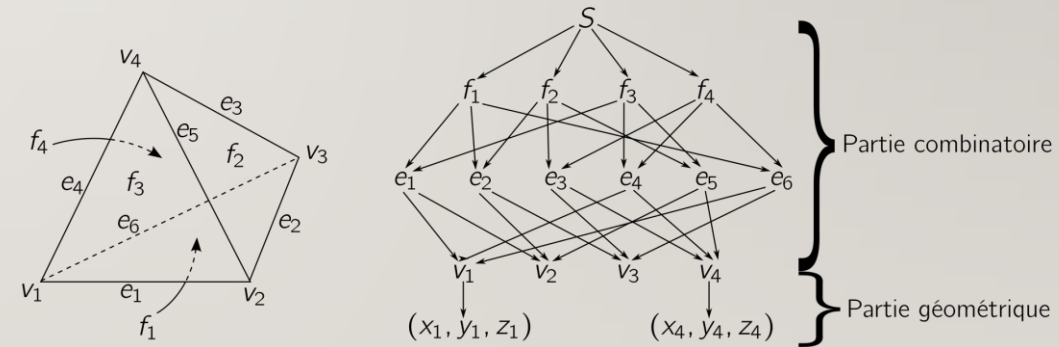
# TELL WATCH

---

- Démonstration Union / Intersection
- Démonstration Montre Tell

# REPRÉSENTATION PAR FRONTIÈRES (BOUNDARY REPRESENTATION B-REP)

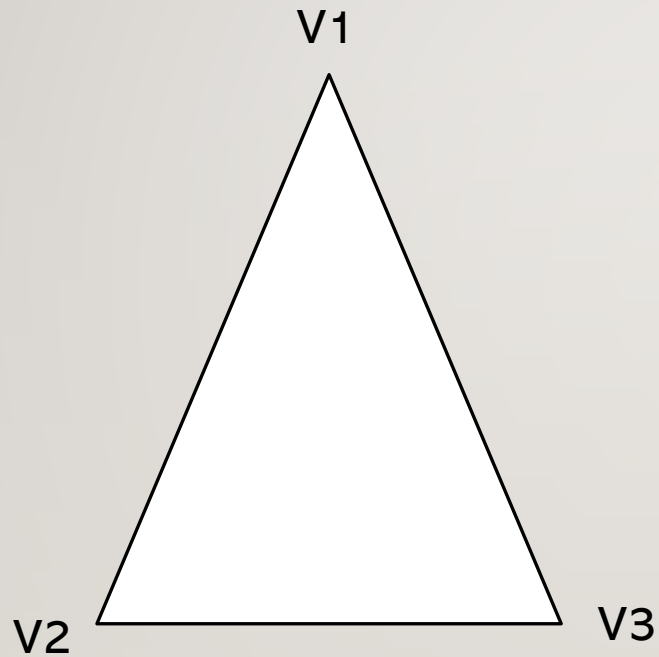
- La surface de chacun des objets est limitée par des formes géométriques simples polygones, Surface de Bézier, B-splines, NURBS, etc....
- Le volume n'est plus plein mais creux et représenté par son bord uniquement
- Pour permettre de dessiner les objets ceux-ci doivent être projetés sur un plan 2D.





# LE TRIANGLE

---



- Élément de base d'une representation B-Rep Classique
- Composé de trois sommets ou vertex en anglais et de trois arêtes ou edge en anglais
- Chaque sommet a une coordonnée dans l'espace mais peut également avoir d'autres attributs, normal, uv, tangente, couleur, matrices (bones)



# TESSELLATION

---

Transformer un objet en ensemble  
de triangle

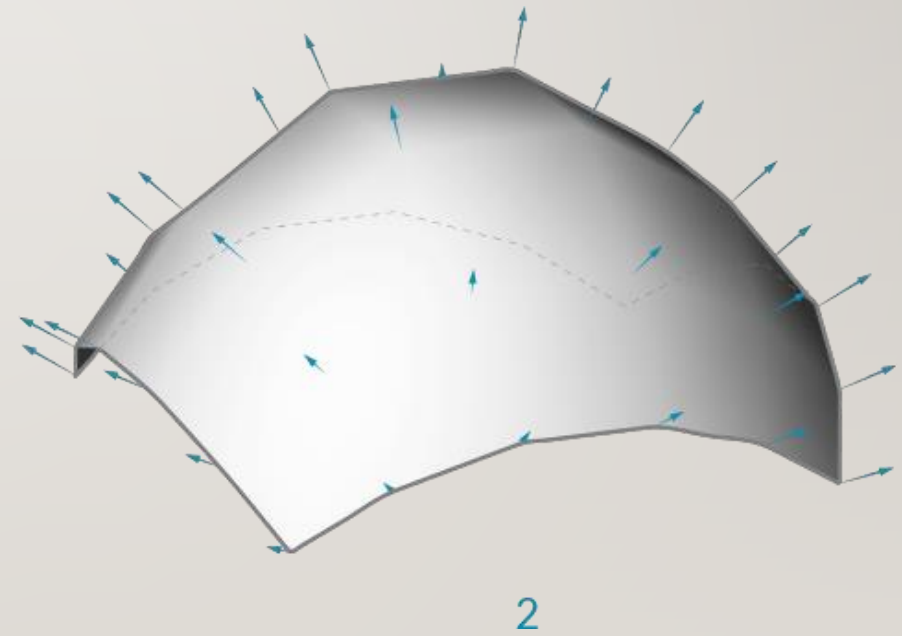
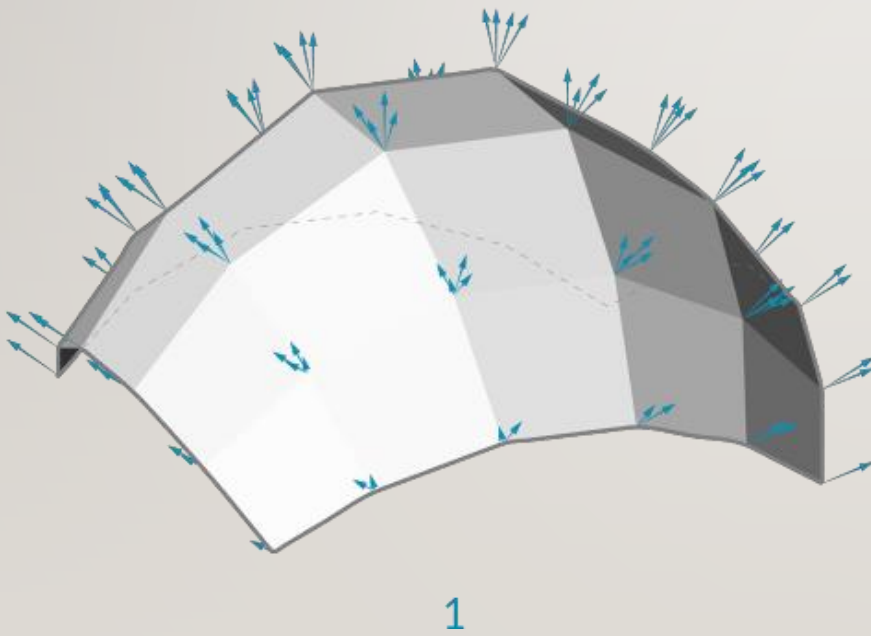
<https://playground.babylonjs.com/>

```
const createScene = function () {  
    const scene = new BABYLON.Scene(engine);  
    const camera = new BABYLON.ArcRotateCamera("Camera", Math.PI / 2, Math.PI / 2, 4  
, BABYLON.Vector3.Zero(), scene);  
    camera.attachControl(canvas, true);  
    const light = new BABYLON.HemisphericLight("light", new BABYLON.Vector3(1, 1, 0)  
, scene);  
  
    const cone = BABYLON.MeshBuilder.CreateCylinder("cone", {  
        diameterTop: 0,  
        tessellation: 100  
    });  
    cone.convertToFlatShadedMesh();  
  
    const sphere = BABYLON.MeshBuilder.CreateSphere("sphere", {  
        segments: 5,  
        diameter: 1  
    });  
    sphere.convertToFlatShadedMesh();  
  
    const cylinder = BABYLON.MeshBuilder.CreateCylinder("cylinder", {  
        tessellation: 5  
    });  
  
    cylinder.convertToFlatShadedMesh();  
    return scene;  
};
```

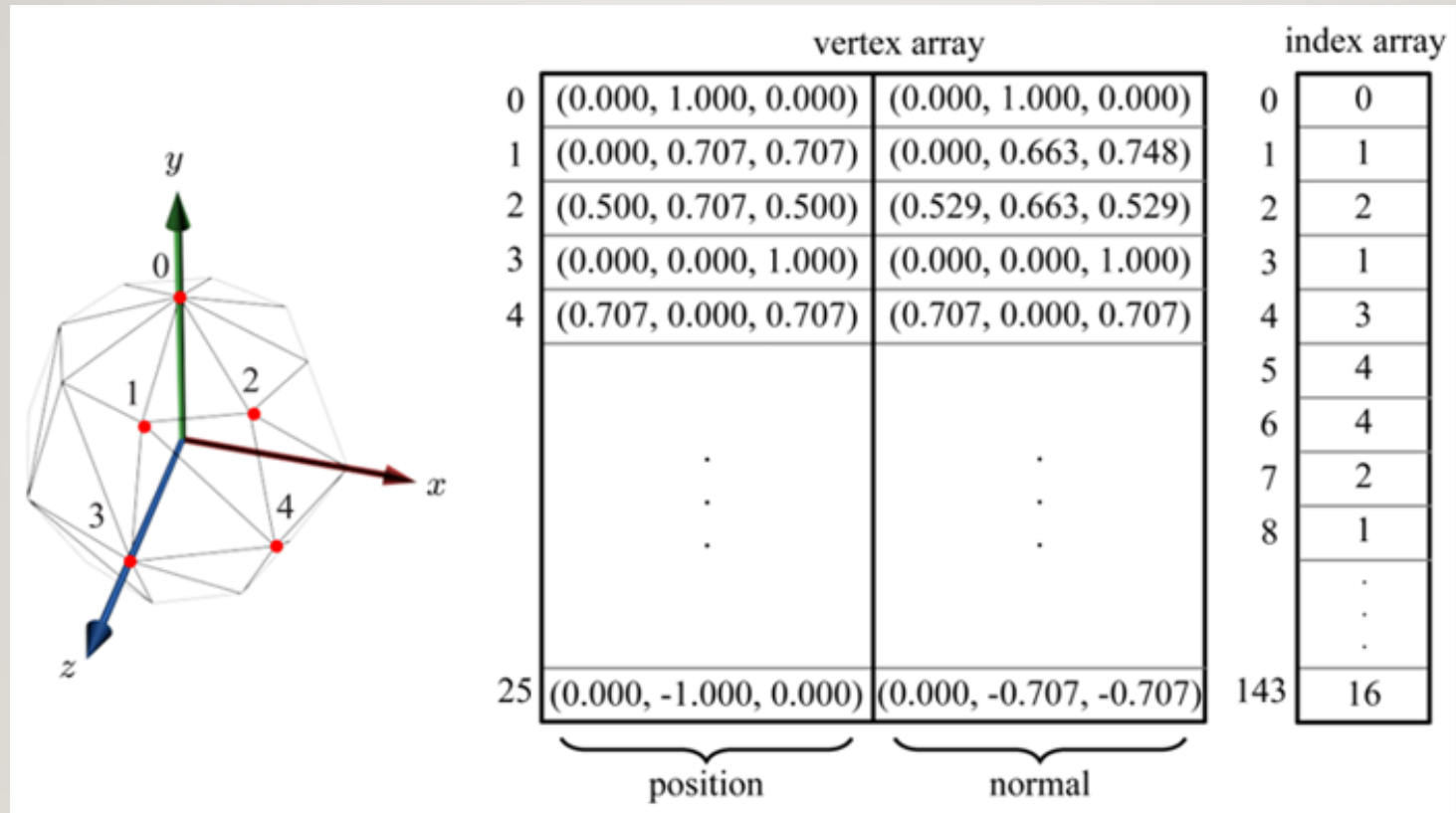
# LES NORMALS

---

- Pour calculer l'illumination de la surface d'un objet on utilise la normal



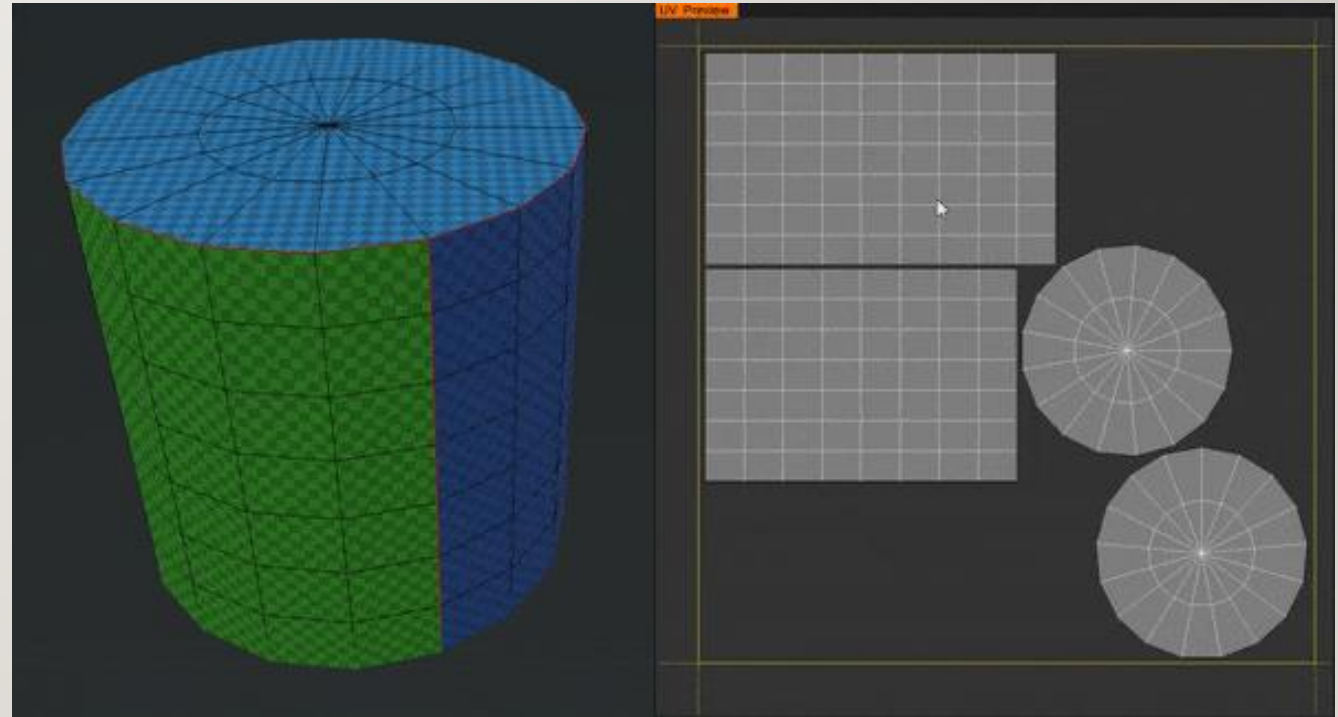
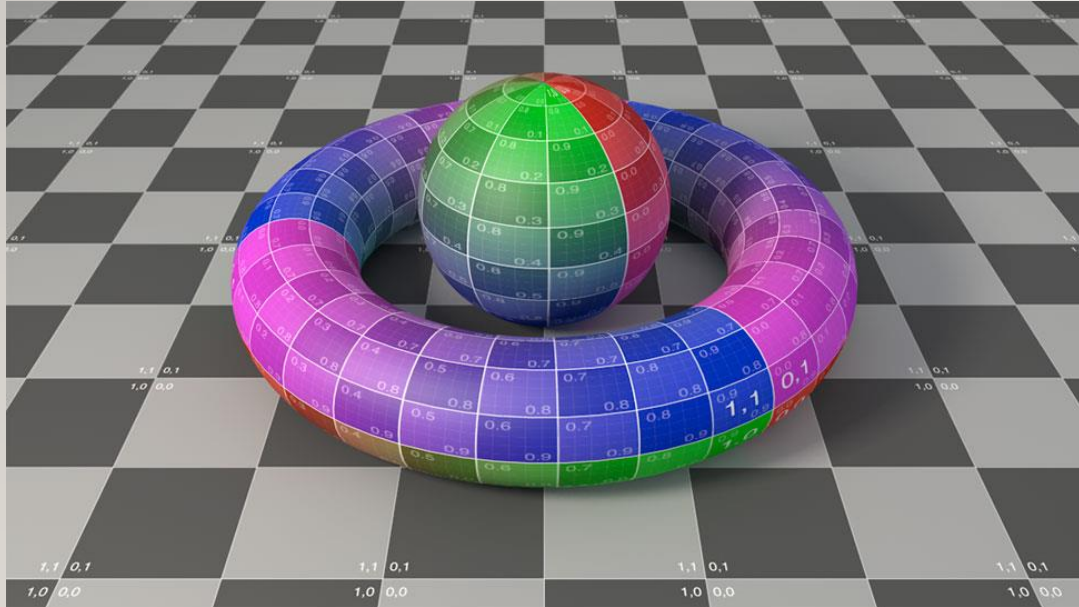
# INDEX, POSITIONS ET NORMALS





# LES COORDONNÉES DES TEXTURES

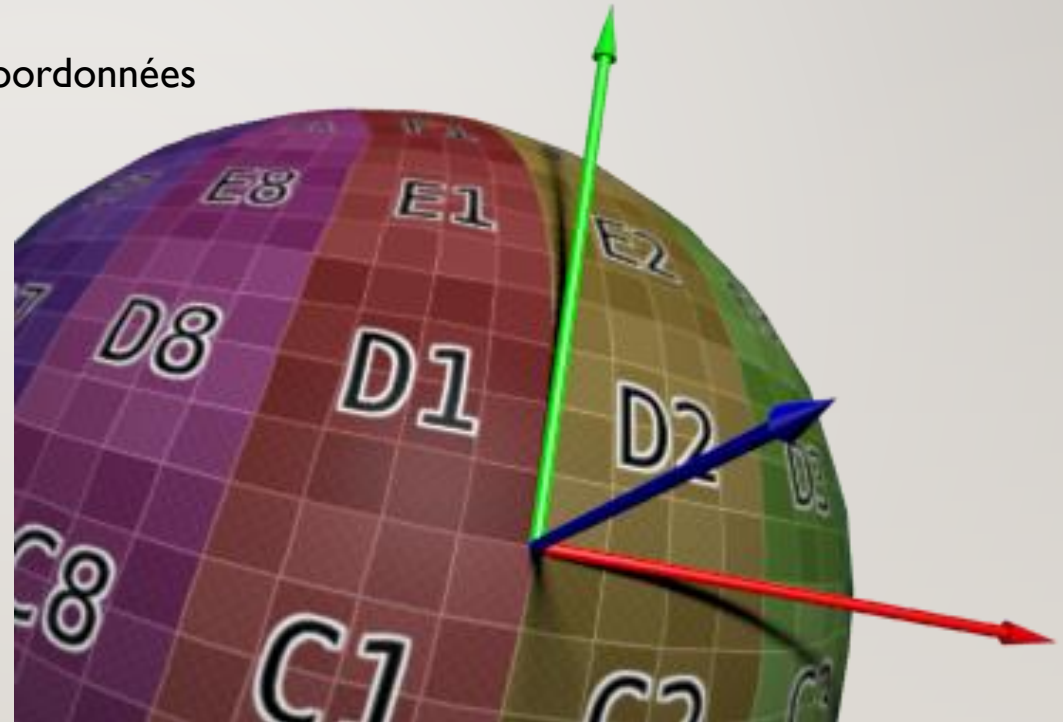
---



# TANGENTE ET BITANGENTE

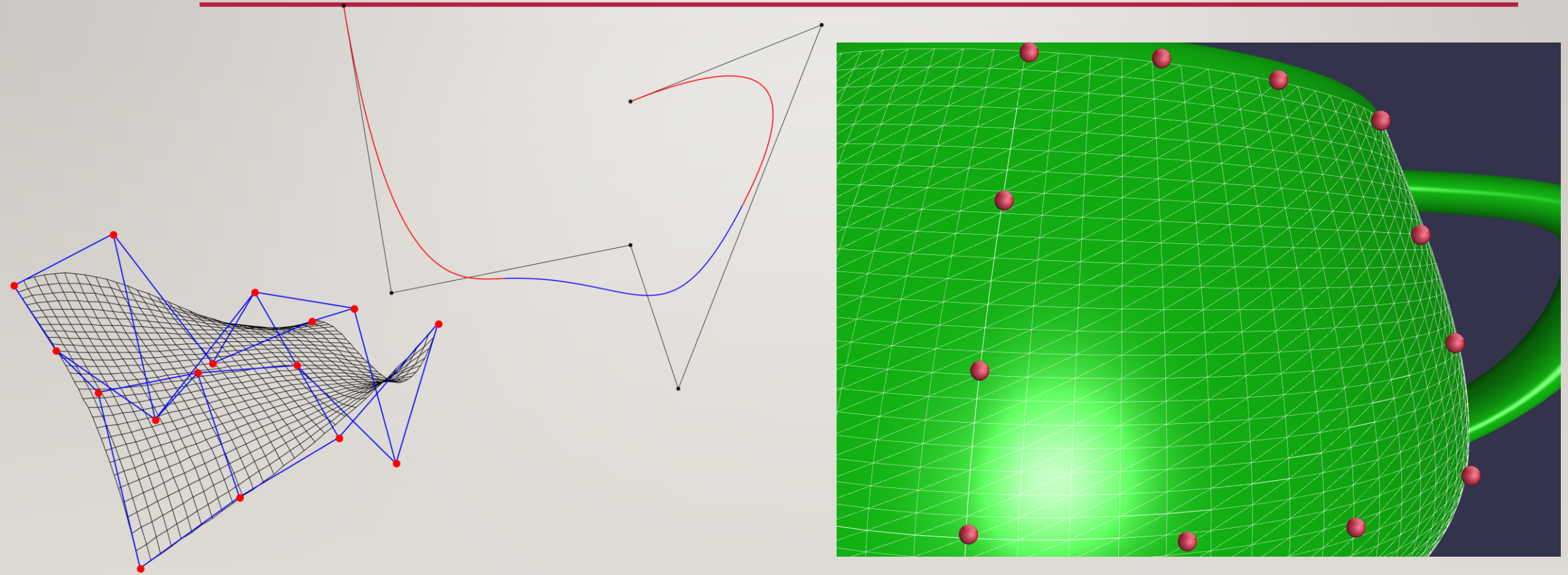
---

- Tangente et Bitangente calculée en utilisant les coordonnées UV





# SURFACE DE BEZIER



# EXERCICE THEIERE SURFACE DE BEZIER

---

- Calculer un nombre suffisant de points sur la surface de Bézier de la théière pour avoir une bonne tessellation
- Créer un maillage à partir de ces points
- Découper les carrés du maillage en deux triangles pour remplir le tableau d'index et de positions et assurer un rendu correct

<https://playground.babylonjs.com/#3DS8RM#2>

Compléter le code ou se trouve les commentaire TODO:

