

Sensibilisation à la programmation multimedia

Christophe Vestri

Le vendredi 25 septembre 2025

Plan du cours

- 1^{er} TD: Intro, github, carto/geo, leaflet/mapBox, rest Api
- 2em TD: 2D/3D: Canvas, WebGL et Three.js/babylon.js
- 3em TD: Three/babylon.js + Leaflet.js cartographie
- 4em TD: IA

Objectifs du cours:

- Bases de géolocalisation et de la cartographie
- Initiation multimédia: 2D/3D, carto/géo et infographie
- Expérimenter quelques méthodes et outils web geo/3D
- Réaliser un petit projet (combinera ce qu'on a vu)

Travail demandé

Sur l'utilisation de l'IA et travail demandé:

- Commencez par lire les docs, comprendre avant de se lancer sur chatGPT.... Partez plutôt des docs pour apprendre
- Vous pouvez discuter, vous entraider mais le travail et le rendu sont individuels
- Le dernier TD sera full IA, à utiliser pour coder ou dans le projet (rendu toujours individuel)

Evaluation:

- 25% par rendu de TD (et aussi participation au TD)
- N'oubliez pas de commiter, **pusher sur Github**
- **Vérifiez le fonctionnement sur Github et Smartphone**
- **Je corrige le soir ou le lendemain soir**

Plan Cours 2

CM

- Rappel dernier cours
- WebGL
- CG et three.js

TD

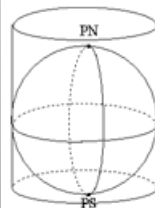
- Three.js (exercice – 1h30)
- Babylon.js (exercice – 1h30)

Géo + Html5 + LeafLet.js

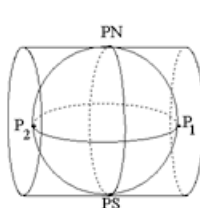
- Repères Géo et carto
- Acces capteur caméra: Géolocalisation, DeviceOrientation, DeviceMotion
- [Leafletjs](http://dmitrybaranovskiy.github.io/leaflet-js/), Mapbox, mapQuest
- Données géolocalisées (REST API)



Représentation cylindrique :

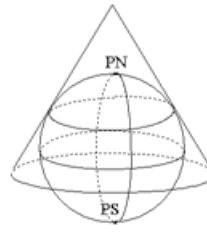


directe

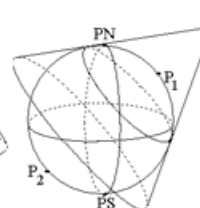


transverse

Représentation conique :



directe



oblique



Graphique en HTML

- Canvas 2D (dessin 2D)



- SVG: Scalable Vector Graphics



- CSS3D: pour des effets de rendu 3D
- WebGL: pour de la 3D basique
- Three.js ou Babylon.js pour de la vrai 3D

WebGL



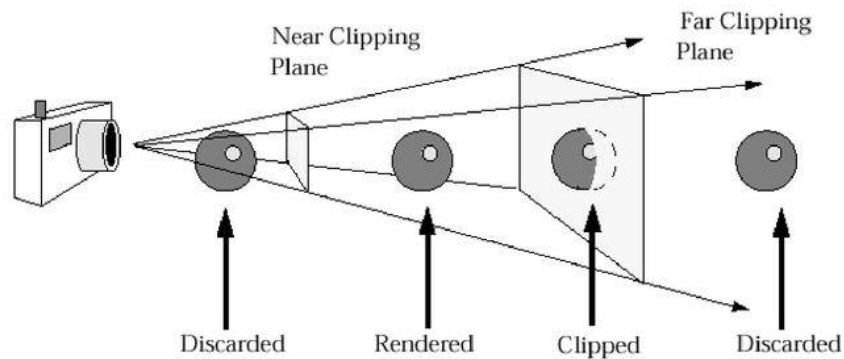
- **Qu'est-ce que WebGL**
 - Cross plateforme et libre de droits
 - OpenGL ES (OpenGL simplifié pour l'embarqué) dans le Web (HTML5)
 - Bonne intégration Html et mécanisme d'évènements
 - DOM API pour affichage 2D et 3D
 - Langage de type script (pas de compilation)
 - Accélérations matérielles et GPU (GLSL)

WebGL

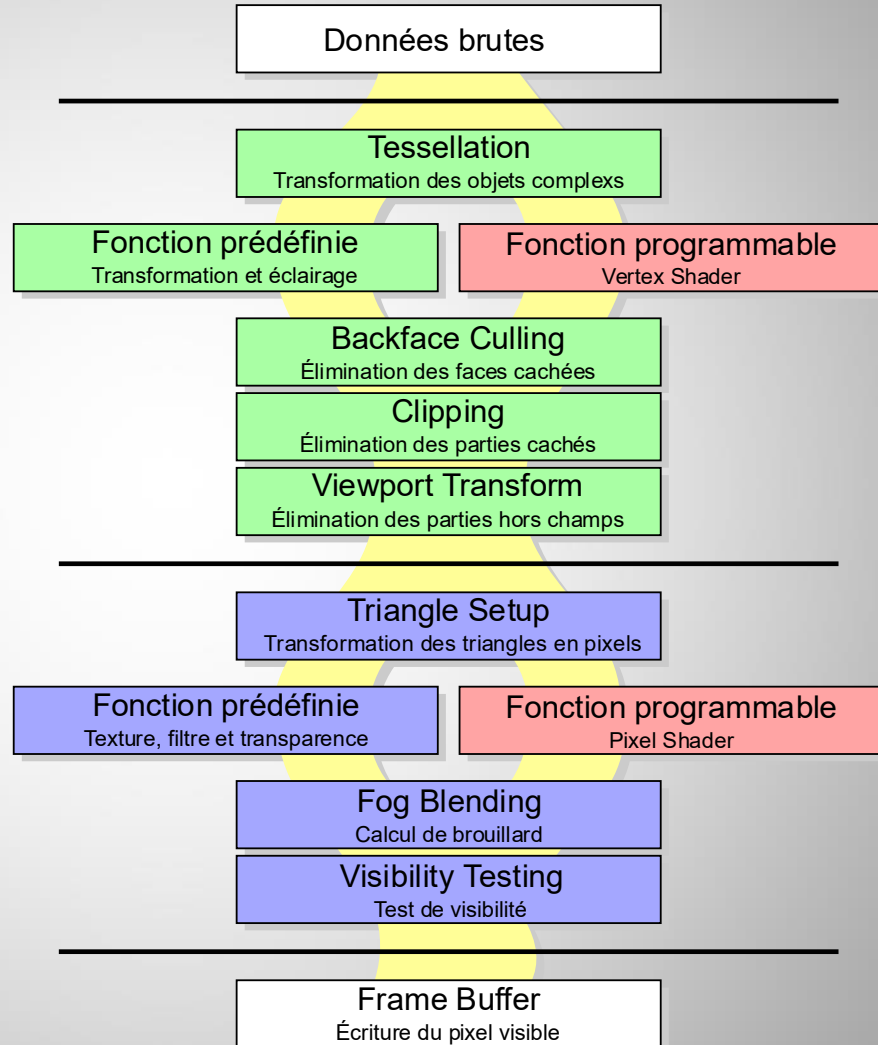
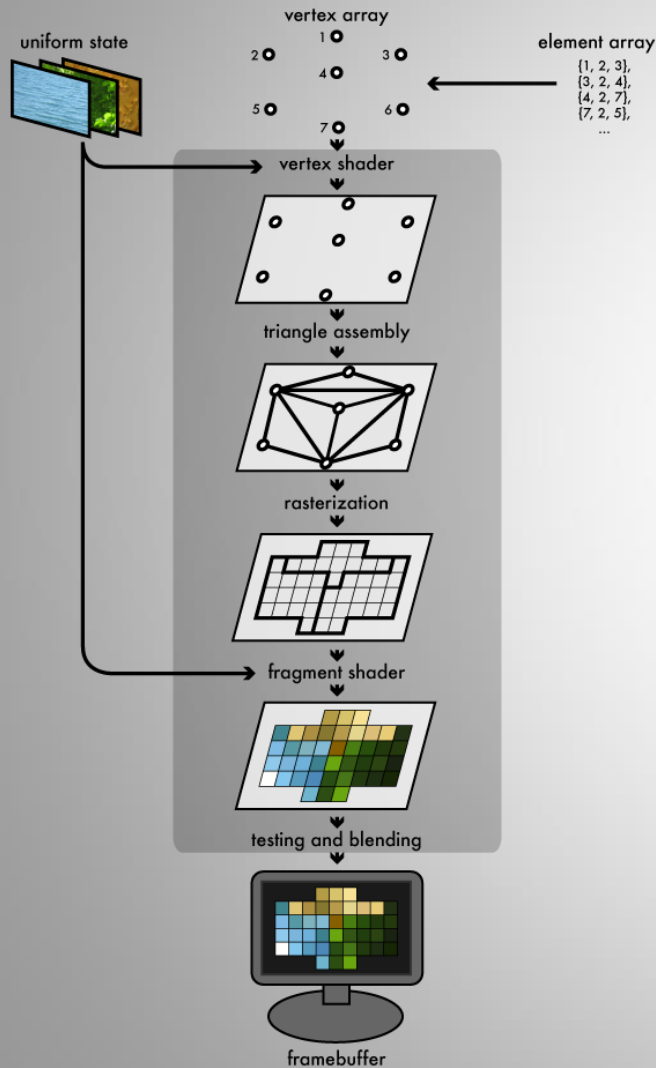
- **Computer graphics**

3D Clipping

- Objects that are partially within the viewing volume need to be clipped – just like the 2D case



WebGL Pipeline

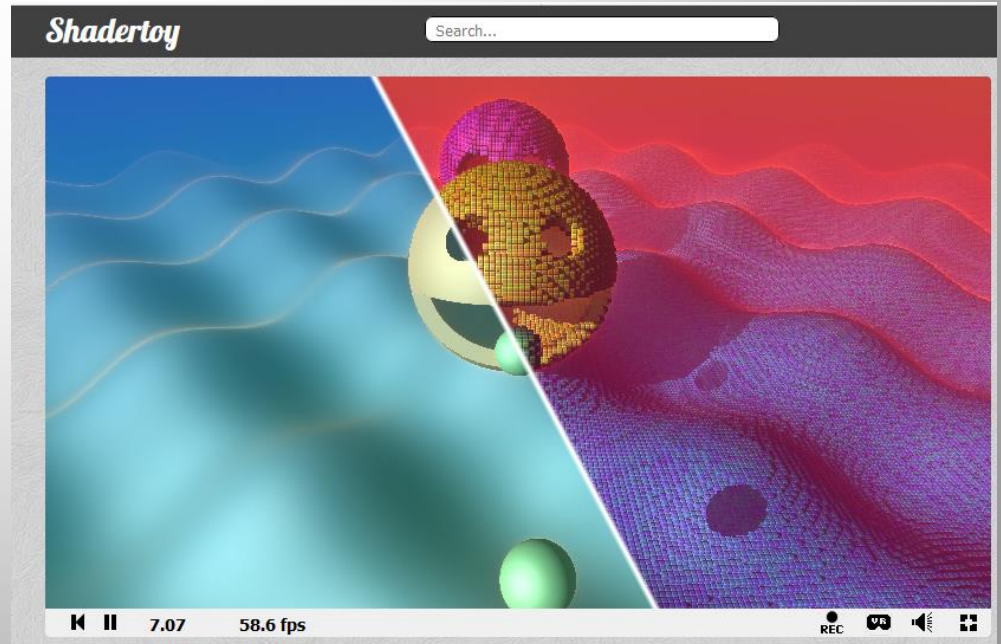


WebGL

- **Low-level API**
 - GLSL OpenGL Shading Language
 - Machine d'état: OpenGL Context
 - Calcul de matrices et transformations
 - Buffers de vertex: positions, normals, color, texture
 - Depth buffer, Blending, transparency
 - Lighting, Cameras...
 - https://developer.mozilla.org/fr/docs/Web/API/WebGL_API
 - <https://webglfundamentals.org/webgl/lessons/fr/>

WebGL Examples

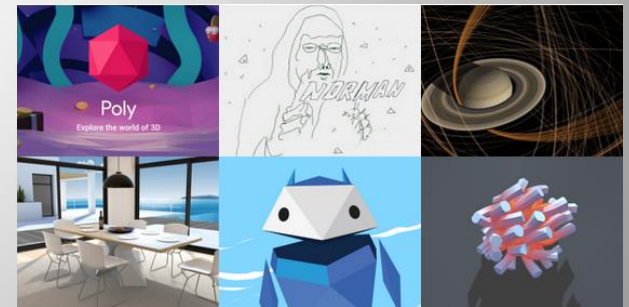
- <https://www.khronos.org/webgl/wiki/Tutorial>
- <https://webglfundamentals.org/>
- <https://www.shadertoy.com/>



Three.js

The logo for Three.js, featuring the word "THREE" in white serif font and ".js" in a smaller white sans-serif font, both on a red rectangular background.

- Qu'est-ce que Three.js
 - Couche abstraite et haut niveau de WebGL
 - Librairie javascript pour créer des scènes 3D
 - Cross-plateforme et gratuit
 - Rendus en webGL, CSS3D et SVG
 - <https://threejs.org/>



Fonctionnalités

THREEJS

- Scenes, Cameras, Renderer,
- Geometry, Materials, Textures
- Lights, Shadows
- Shaders, Particles, LOD
- Loaders: Json compatible Blender, 3D max, Wavefront OBJ, Autodesk FBX
- Animation, Trackballcontrols, Math Utilities
- <https://davidlyons.dev/threejs-intro/>

Plan Cours 2

CM

- Rappel dernier cours
- WebGL
- CG et three.js

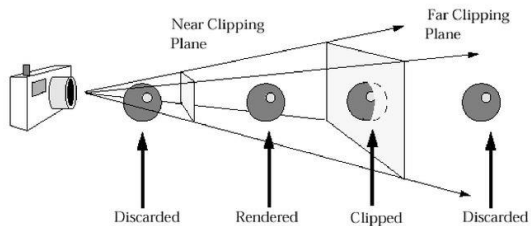
TD

- Three.js (exercice – 1h30)
- Babylon.js (exercice – 1h30)

Rappels avant exercices

3D Clipping

- Objects that are partially within the viewing volume need to be clipped – just like the 2D case

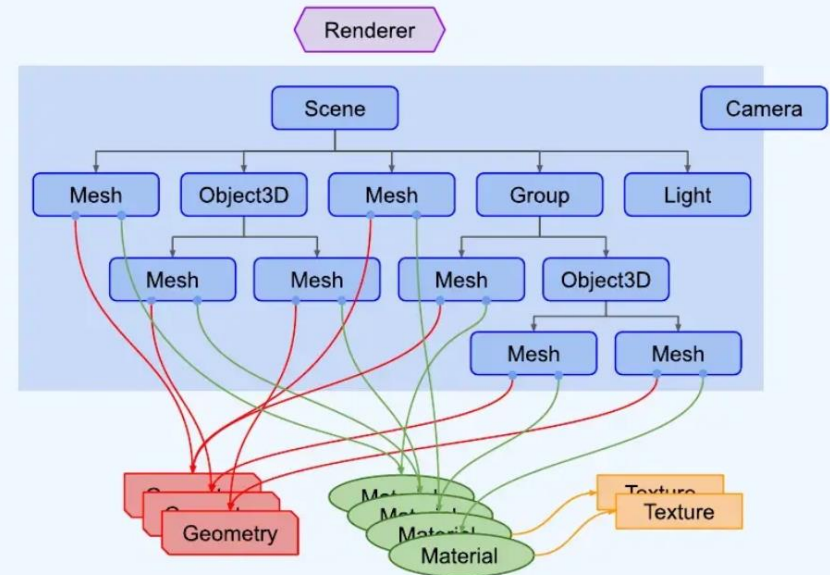


September 15, 2018

Computer Graphics

6

Attention au 3D Clipping



Il vous faut :

- 1 scene
- 1 camera
- 1 light
- Un renderer

Les principaux problèmes

1. Scene mal éclairée (éclairage directif):
 - Solution: éclairage ambiant pour commencer
2. Objet géométrique non visible
 - Choisissez une position de caméra, placer l'objet devant
 - Faites 1 dessin sur papier pour être sûr de ce que vous faites
 - Problème de clipping?
3. Mon modèle 3D ne s'affiche pas:
 - Vérifiez la console de votre navigateur (les erreurs...)
 - Enlevez la texture, mettez un matériau simple
 - Vérifiez l'échelle de votre objet et les positions (voir 2)
 - Utilisez un serveur local (slide précédent)
 - Utilisez un modèle glTF des exemples de Three.js avant d'utiliser le votre
4. Mon objet ne bouge pas
 - Vérifiez que vous appelez bien : `renderer.setAnimationLoop(animate);` ou `engine.runRenderLoop(renderLoop);`
 - Il doit y avoir une variable (angle/position/scale) qui varie, testez avec un breakpoint

Outils de debug

- En local (besoin pour charger modèles 3D):
 - Avoir python (miniconda ou autre)
 - Se placer dans le répertoire html
 - `python3 -m http.server`
 - <http://localhost:8000/> firefox ou chrome
- Smartphone android -> Chrome
- <https://developers.google.com/web/tools/chrome-devtools/javascript>
 - Simulation de smartphone (F12)
 - Connecté à un smartphone: <chrome://inspect/>

Exercice 1 – Three.js

- Exercice 2 (1h30) <https://github.com/vestri/Initiation-multimedia>
 - <https://threejs.org/docs/index.html#manual/en/introduction/Installation>
 - <https://threejs.org/docs/index.html#manual/en/introduction/Creating-a-scene>
- Objectif: testez les éléments de base d'une scène 3D:
 - Créez une scène + caméra + light + renderer
 - Créez un objet générique (sphère ou cube)
 - Texturez cet objet
 - Téléchargez un objet 3D
 - Animez un objet avec le smartphone, DeviceEvents, DeviceOrientation et/o DeviceMotion
 - Ajoutez ce que vous voulez: Fog/pluie ou particules
- bonus: mettre un contexte

Chaque fonctionnalité
donne des points

Publiez sur votre Github
pour que je puisse corriger

- **Qu'est-ce que Babylon.js**
 - **Idem Three.js**: librairie javascript pour créer des scènes 3D, cross-plateforme et gratuit
 - **avec d'autres fonctionnalités**: Gaussian Splat Rendering, Physics, WebXR, Apple Vision Pro...
 - <https://www.babylonjs.com/>
 - <https://doc.babylonjs.com/setup/starterHTML/>
 - <https://doc.babylonjs.com/journey>



Exercice 2 – Babylon.js

- Exercice 2 (1h30)
 - <https://doc.babylonjs.com/journey>
- Objectif: testez les éléments de base d'une scène 3D:
 - Créez une scène + caméra + light + renderer
 - Créez un objet générique (sphère ou cube)

même chose que Three.js
 - Texturez cet objet
 - Téléchargez un objet 3D
 - Animez un objet avec le smartphone, DeviceEvents, DeviceOrientation et/ou DeviceMotion
 - Ajoutez ce que vous voulez: Fog/pluie ou particules
- bonus: mettre un contexte

Chaque fonctionnalité
donne des points

Publiez sur votre Github pour que je puisse corriger