M2-Images

pipeline grahique et openGL

J.C. Iehl

September 19, 2016

résumé des épisodes précédents

intro pipeline openGL :

- 2 parties : transformation des sommets et couleur des fragments,
- présentation de l'api,
- application test : dessiner 1 ou quelques triangles.

mon premier shader

vertex shader :

- pas de transformation,
- coordonnées des sommets dans le volume visible,

fragment shader :

- couleur constante,
- peut mieux faire...

cf tuto3GL.cpp et tuto3GL.glsl, ou tuto2GL.cpp

exemple :

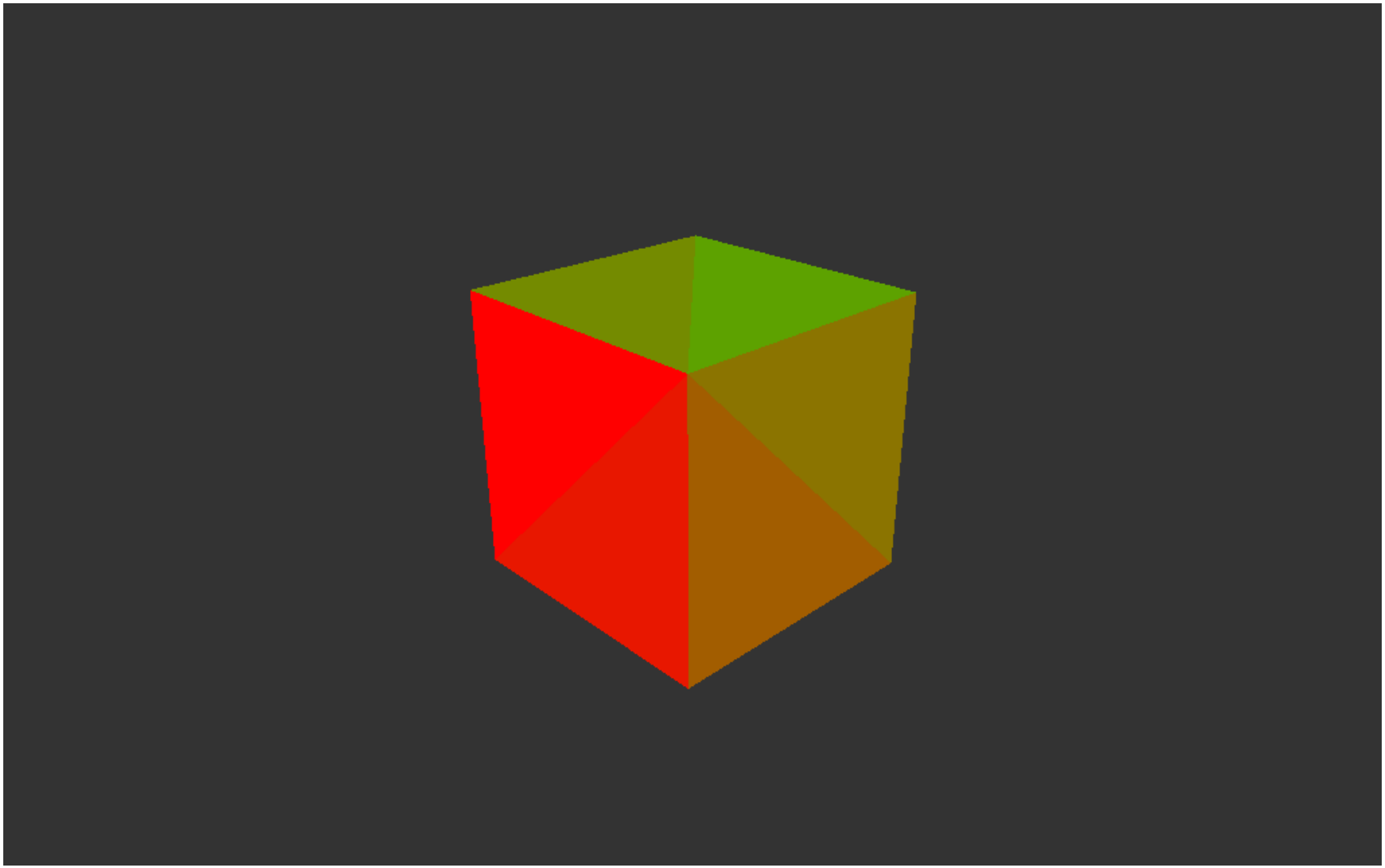
mon premier shader

question 1 :

donner une couleur différente à chaque triangle ?

rappel : int gl_PrimitiveID

exemple :



mon premier shader

question 2 :

et si on pouvait bouger la camera / l'orientation du cube ?

rappel : glGetUniformLocation(...);
glUniform(...); glUniformMatrix(...);

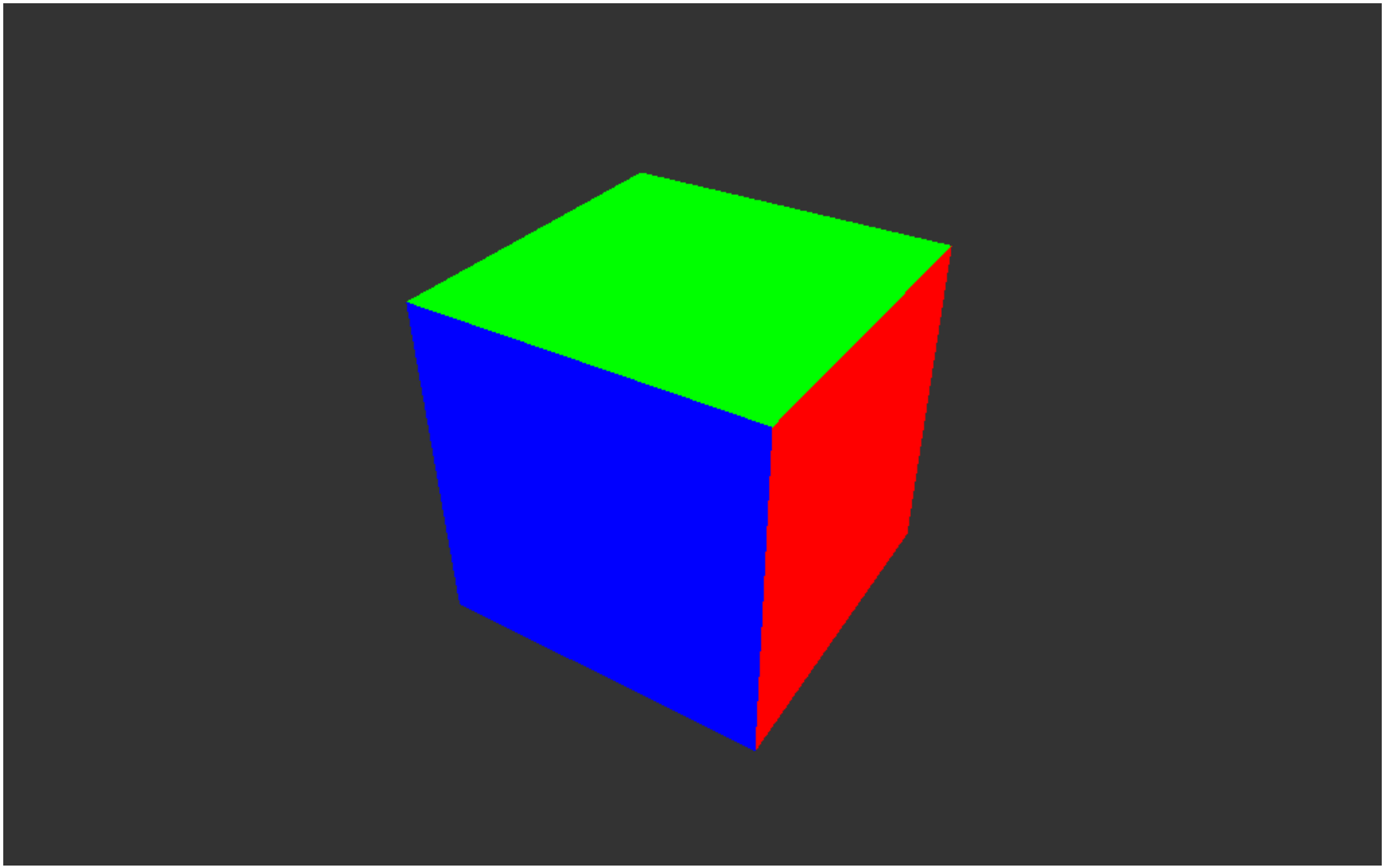
mon premier shader

question 3 :

donner une couleur différente à chaque triangle...
en fonction de la normale de la face ?

rappel : gl_VertexID, varyings...

exemple :



et en fonction de la normale...

indication :

- la normale d'un triangle :
- cross(ab, ac) avec ab et ac les 2 aretes issues de a.

connaissant l'indice d'un sommet, déterminer l'indice des 3 sommets du triangle...

et en fonction de la normale...

indication :

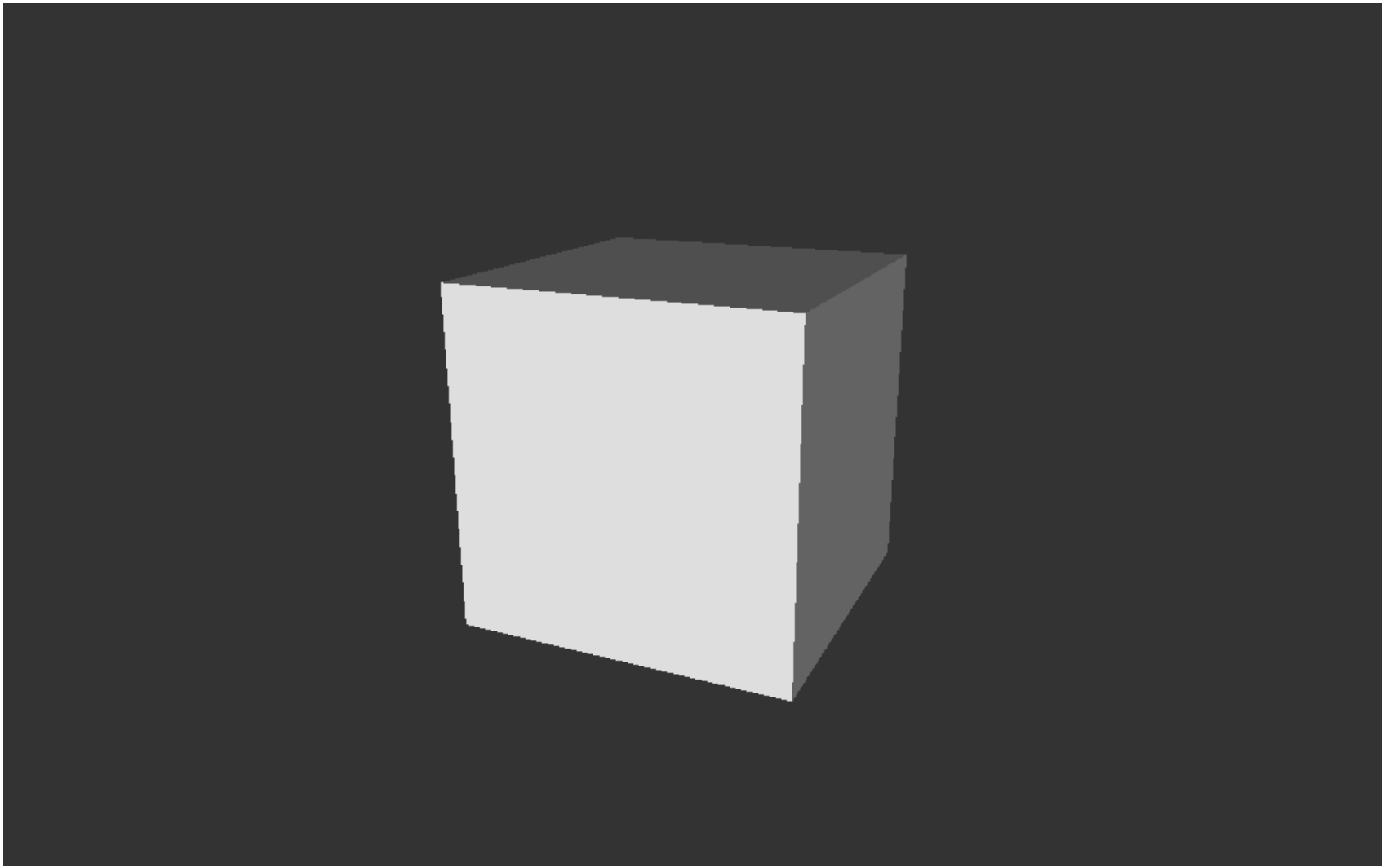
- transférer les normales en plus des positions des sommets...
- déclarer un autre tableau d'uniforms,
- lui affecter les normales des sommets...

mon premier shader

question 4 :

et en fonction l'orientation de la face par rapport à la camera ?

exemple :



mon premier shader

question 5 :

et en fonction de l'orientation de la face par rapport à un point quelconque ?
(une source de lumière)

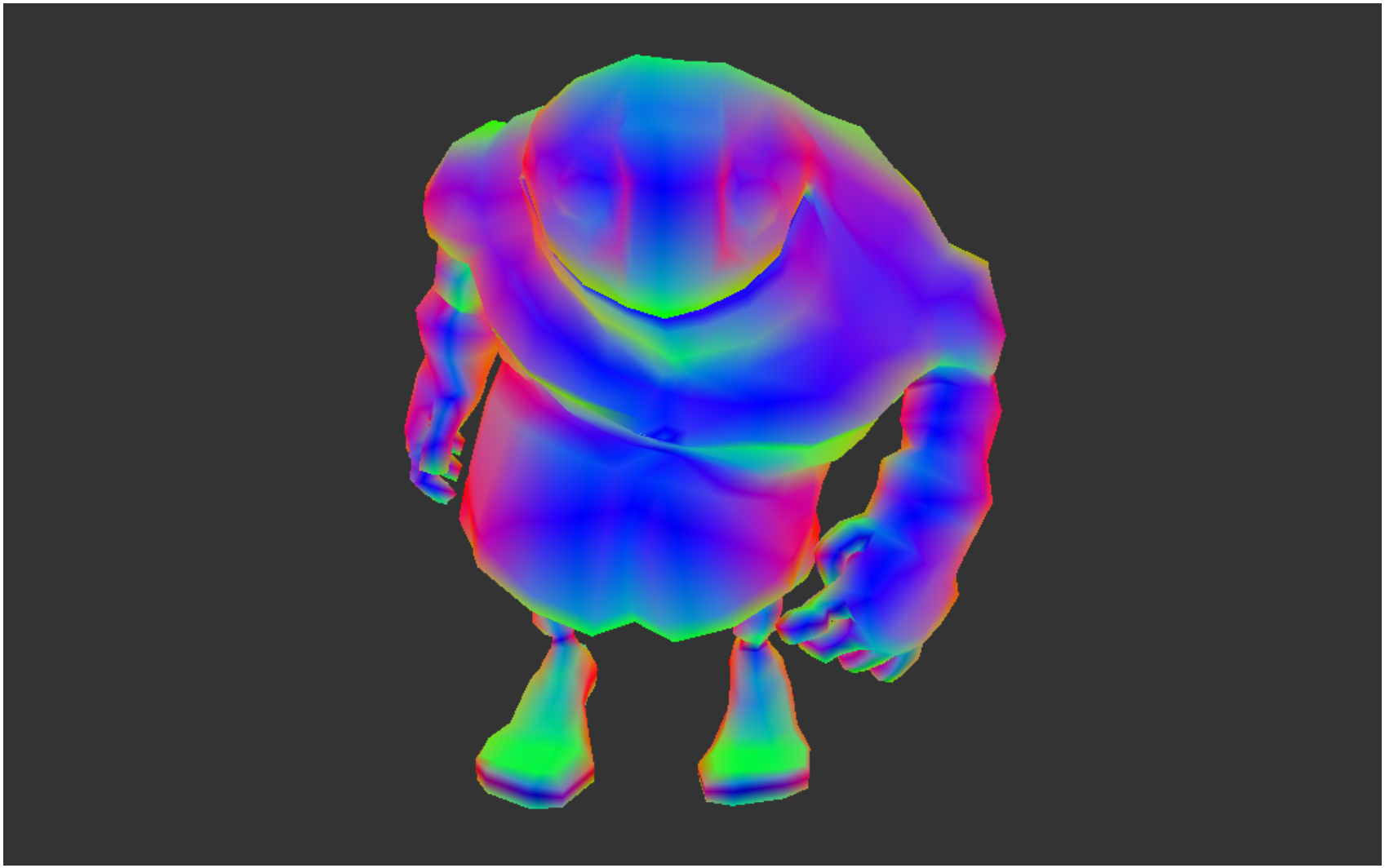
bonus : changez la position de la source en fonction du temps...

mon premier VAO

question 6 :

- déclarez les positions et les normales comme 2 attributs de sommets,
- dans le shader...
- que faut-il modifier dans l'application ?

exemple :



exemple :