Email : [romain.vergne@iniria.fr](mailto:romain.vergne@iniria.fr)

http://romain.vergne.free.fr/teaching/webgl/cours01.pdf

Diapo 2 :

On fait des images pour qu’elles soient vu par les humains.

Fonction qui dépends des modèles, matériaux, lumière qui donne une image :

Fonction : f(F,M,L) -> I

Impossible à partir d’une image les propriétés physiques qui ont découlé d’une image (DONC I-> F,M,L impossible, le cerveau fait des estimations).

Diapo 30

Norme d’un vecteur = longueur

* Se calcule avec la distance x² + la distance y² + la distance de z²
* On met au carré car distance forcément positive, mais du coup après on fait racine carrée du résultat

Si on divise un vecteur par sa norme, on a une taille de 1. Cela sert en modélisation on s’en fout de la longueur, on veut juste le point

Produit scalaire = norme de A \* norme de B \* cos(vecteur A et B) = cos(vecteur A et B) = donne l’angle entre les vecteurs

Ex : Si vecteurs même direction, longueur, angle = 1 car cos(0) = 1

TP1 :

Plane.obj = avion modélisé

* Si on veut use un objet qu’on a blendait, exporter en obj

Pour représenter triangle tableau avec tous les sommets [P1,P2,P3,P4] = attribut

Puis un autre tableau qui va pointer (donc composé d’indice) sur le tableau précédemment pour avoir les points nous intéressant, ex : [0,1,2,2,1,3]

On peut faire aussi un tableau avec les infos de la normes ex : [n1,n2,n3,n4]

Pour les missiles : vertices pour position sommet, coords pour la texture

Fonctionnement carte graphique

Vertex shader

Mémoire

Rastérisation

GPU

FB

Fragment shader

Notre maillage est constitué de sommet et triangle. Le vertex shader est lui qui va appliquer une transformation sur ce qui lui est donné en entrée. L’idée est donc de faire la même transformation sur chacun des sommets.

gl\_position = variable pour la position finale du sommet sur l’écran. Vecteurs contenant 4 flottants car quand on fait du 3d, on utilise des matrices 4x4.

Rasterisation = Convertit donnée vectiorielle en pixel, bitmap. On vérifie emplacement et on le colore.

Fragment shader/Pixel shader = Il va récupérer toutes les informations d’avant (position, coordonnées) et va appliquer toutes les fonctions qui veut pour avoir la couleur demandé.

Il faudra un shader par élément de dessin. Si par exemple même donnée d’entrée pour plusieurs objets différents, on utilisera un shader pour plusieurs objets.

aVertexPosition = position du sommet courant

On a des coordonnées à l’intérieur des pixels