MoodleExamen (Génies informatique & logiciel)

INF2705 - Infographie

Commencé le	mardi 17 octobre 2023, 08:30
État	Terminé
Terminé le	mardi 17 octobre 2023, 10:29
Temps mis	1 heure 59 min
Note	**24,00** sur 30,00 (**80**%)

Description
Marquer la question

L'examen comporte 10 questions pour un total de **30 points**.

Vous avez droit à **une feuille de notes recto verso** ainsi qu'à une calculatrice non programmable.

Il n'est pas nécessaire de réécrire la question dans votre réponse. Par exemple, à la question « *Quelle est la librairie graphique que nous avons utilisée?* », il suffit de répondre « *OpenGL* » plutôt que réécrire « *La librairie graphique que nous avons utilisée est OpenGL* ».

- Une question qui débute avec « Différenciez... » ou « Dites la différence entre ... » demande de donner la différence entre des concepts. Il faut généralement expliquer brièvement chaque concept et en faire ressortir ensuite les différences.
- Lorsqu'il est dit « expliquez la méthode ou l'algorithme », on s'attends à une explication du « comment ».
- Lorsqu'il est dit « qu'est-ce que cette chose », il faut donner une *description* de cette chose et non à quoi elle sert.
- Dans vos réponses, assurez-vous de montrer que vous comprenez la matière et que vous comprenez ce que vous écrivez. Une réponse floue, trop vague ou imprécise ne donnera pas de points
- Si je ne demande pas de justifier, il est **inutile** de justifier. Vous pouvez le faire, mais attention à votre gestion de temps et dites vous que chaque ligne de texte en trop est une occasion de perdre des points.

Bonne chance!

NAVIGATION DU TEST

i 1 2 3 4 5 6 7 8 9 10

Afficher une page à la fois

Terminer la relecture

Question 1
Terminé
Note de 1,00 sur 1,00
Marquer la question

Lorsque des objets projetés sont moins larges qu'un pixel, il est possible d'observer une certaine fragmentation de l'image le long des arêtes des objets, ce qui rends l'image inexacte. Quelle technique résout ce problème.

- (●) Anticrénelage
- ( ) Rastérisation
- ( ) Ajustement de la coordonnée homogène
- ( ) Interpolation des vecteurs normaux

Note de 1,00 sur 1,00

Question 2
Terminé
Note de 2,00 sur 2,00
Marquer la question

Ayant tellement apprécié votre cours d'infographie, vous décidez d'accepter un stage chez Ubisoft Montréal dans le département de modélisation 3D.

Comme rite d'initiation, votre team lead vous dit que vous serez en charge d'une simple simulation de fumée pour un projet secret.
Malheureusement, la librairie graphique utilisée n'est pas OpenGL, mais plutôt une solution maison. Par contre, vous êtes débrouillard et décidez d'implémenter une solution temporaire dans les nuanceurs qui ressemble étrangement aux nuanceurs GLSL vus en cours.

1. Comment s'appelle l'opération sous openGL pour simuler un effet de fumée dans notre scène?
2. Décrivez brièvement comment vous pourriez arriver à un résultat presque identique à l'aide des nuanceurs de sommets et fragments.

1- Il s'agit de l'opération mix (je suppose la qu'un mix de couleur peut creer une sorte de floutage avec un certain degré de min ou de max des couleurs/textures)

2- dans le nuanceurs de sommet, je pourrais recuperer la position et la normamlisée comme a l'habitude en multipliant la position par une matrice mvp afin d'avoir la position de sommet a l'ecran,

dans le nuanceur de fragment, on pourrais prendre en entrée un uniforme pour recuperer la texture et la retransmettre en sortie. afin de fusionner la texture de fumée a la plecedente on pourrait envoyer en sortie la somme des deux tectures (faire un textcoordout=clamp(texture(fumée,textcoordin)) + texture (actuel, textcoodin)) afin d'avoir un effet de fumée en supposant qu'on a une image de fumée qu'on veut assembler a notre image actuelle pour faire une effet de fumée (je suppose aussi que la fumée ne se déplace pas)

Question 3
Terminé
Note de 1,50 sur 2,00
Marquer la question

Expliquez ce qu'est la technique du mipmapping et quel problème des pixels/texels elle résout.

Le mipmap est en soit une série de textures préalablement filtrées ayant des résolutions assez décroissantes.
En fait, elle permet d'adapter les textures aux dimensions afin de fournir des rendus beaucoup plus beau .

Commentaire :
"ayant des résolutions *assez* décroissantes" Que voulez-vous dire assez décroissantes?

Question 4
Terminé
Note de 1,00 sur 2,00
Marquer la question

Une sphère est tracée dans une application graphique 3D. Cette sphère est entièrement contenue dans un volume de visualisation qui respecte le rapport d'aspect et elle est entièrement visible dans la fenêtre à l'écran.

a) Si cette application utilise une projection orthographique, est-ce que la silhouette de la sphère à l'écran sera toujours un cercle, quelle que soit sa position à l'écran ? Justifiez

b) Si cette application utilise une projection perspective, est-ce que la silhouette de la sphère à l'écran sera toujours un cercle, quelle que soit sa position à l'écran ? Justifiez.

a) Oui, sa silhouette sera identique quelque soit sa position a l'ecran car les sommets étant projetés parrallellement au plan avant, la projection orthographique préserve la taille des objets.
b) La sphere sera déformée (ne sera pas toujours un cercle) si le rapport du glViewPort() n'est plus respecté

Commentaire :
b) On spécifie que le rapport d'aspect est respecté

Question 5
Terminé
Note de 2,00 sur 2,00
Marquer la question

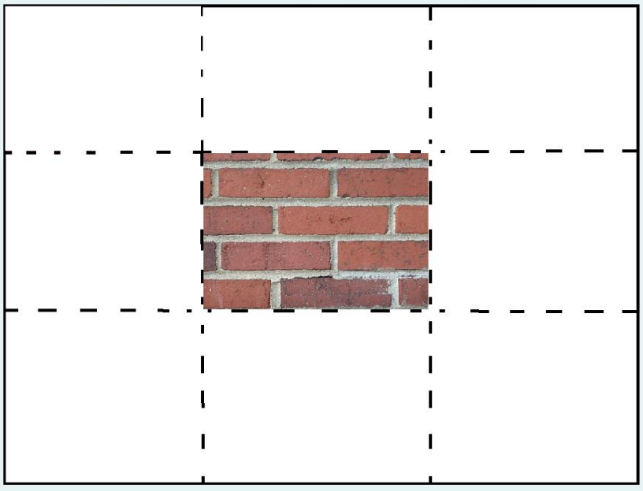
Quelle est la différence entre une image matricielle et une image vectorielle?
Quelles informations sont stockées dans chacun des types d'images?

- Les images matricielles sont représentée par des matrices et celles vectorielles par des vecteurs
- Les informations stockées dans les images matricielles sont des coordonnées de représentation et dans celles vectorielle : données de représentation de l'image et de l'assemblement des divers points/position

Commentaire :

Question 6
Terminé
Note de 1,00 sur 3,00
Marquer la question

Supposons que vous voulez appliquer une texture de briques sur le mur d'une maison qui est représenté par un *quad* dans votre scène .
Toutefois, vous remarquez que votre texture de briques n'est pas d'une grande résolution et le rendu laisse à désirer (Voir l'image ci-bas).
Étant un étudiant allumé, vous vous souvenez d'une certaine caractéristique des textures en OpenGL et décidez de l'utiliser pour vous éviter de trouver une texture d'une plus grosse résolution.
Quel est le nom de cette propriété/caractéristique et quelles seraient les paramètres que vous utiliseriez avec la propriété ainsi que les valeurs des coordonnées de textures pour l'exploiter adéquatement.



je pense qu'il s'agit de glTexParameterf ou glTexParameterif. on peut ajouter le paramettre mag pour avoir une groisissemement de l'image
aussi, un gl_texture_wrap T pour avoir une modification verticale mais aussi horizontale avec gl_texture_wrap_S afin de pouvoir bien l'agrandir dans les deux sens. Enfin, on pourrait utiliser un gl_clamp_to histoie de pas répéter l'image mais de la prendre telle quelle.
pour prendre en compte tout l'espace disponnible, on pouurait donnér au point le plus en bas et a gauche de l'image les coordonnées (0;0)
a celui le plus en bas et le plus a droite (1;0). celui le plus en haut et le plus a gauche (0;1) et le dernier point, celui le plus en haut d el'image et le plus a droite (1;1)

Question 7

Terminé

Note de 2,50 sur 3,00

Marquer la question

Donner les trois matrices nécessaires pour effectuer les transformations suivantes:

1. Une translation vers les X positifs de 3 unités, les Y négatifs de 5 unités et les Z positifs de 3 unités.
2. Une mise à l'échelle qui produira une réflexion de la primitive selon l'**axe des Y** avec **aucun changement** sur la taille de la primitive.
3. Une rotation autour de l'**axe des Z** d'un angle de 30degrés.

Présenter vos matrices sous la forme:

[rangée1 / rangée2 / ... / rangéeN] *avec des virgules ou point virgules entre chaque composantes (chaque colonnes)*

Par exemple une matrice 2x2 : [1 ; 2 / 3 ; 4]

1-[1;0;0;3/0;1;0;-5/0;0;1;3/0;0;0;1]---TRANSLATION --ne peut pas etre représentée en 3D vue qu'il s'agit d'une addition de base qu'on force en produit

2- [1;0;0;0/0;-1;0;0/0;0;1;0/0;0;0;1]

3- [1;0;0;0/0;cos(30);-sin(30);0/0;sin(30); cos(30);0/0;0;0;1/0;0;0;1]

Commentaire :

Question 7 : mauvaise matrice de rotation

Question 8

Terminé

Note de 2,50 sur 4,00

Marquer la question

Objets graphiques sur GPU.

Lors des premiers TPs nous avions utilisé des VBO afin de tracer nos figures. Toutefois, il est possible de tracer une figure sans avoir recours à ces "objets". Répondez aux questions suivantes:

1. Que représentent un VBO et un VAO?
2. Quel est l'avantage principal d'utiliser un VBO pour le traçage d'une primitive et est-il possible de l'utiliser dans d'autres situations? Justifiez.
3. Pourquoi avions-nous mis les vecteurs *position* et *couleur* dans le même VBO de façon à ce que les données soient entrelacées *(pos1, coul1, pos2, coul2, ..., posN, coulN)* au lieu d'utiliser deux VBO distincts?

1-Le VBO (Vertex Buffer Object) est un tableau permettant soit de contenir soit les sommets(afin d'eviter leur répétition/redondance) soit de contenir les connectivités (qui précisent ccomment chaque somment sont reliés)

- Le VAO (Vertex Array Object) conserve les informations nécessaires pour établir le contexte graphique (le 'comment' ou encore tous les details du dessin)

2- Comme expliqué en 1- les VBO permettent d'eviter la redondances des sommets qui sans ce tableau seraient juste répétés de nombreuses fois. Mieux encore il permet distinctement avec le tableau de connectivité d'expliquer les liaison inter-sommet pour permettre une récupération et une utilisation plus facile et efficace de l'information.

3- Cela permet une economie de ressources et rend le code plus simple a utiliser. Je dirais, en soit, que cela optimise le code en matiere de configurations, de 'set up'

Commentaire :

Question 9

Terminé

Note de 4,50 sur 5,00

Marquer la question

Nous avons vu le pipeline graphique programmable de base d'openGL qui est composé d'un nuanceur de sommets et d'un nuanceur de fragments.

Décrivez les données/informations que chacun de ces nuanceurs doit recevoir en entrée, et ce qu'il doit produire *au minimum* en sortie. Détailler également les étapes intermédiaires entre ces nuanceurs (comment un sommet est transformé en fragments).

Finalement, justifiez, en vos mots, l'utilité d'avoir un pipeline programmable en présentant un exemple de situation qui ne serait pas réalisable avec un pipeline standard.

- Le nuanceur de sommet : Il prend en entrée un vecteur qui représente la position dans le monde réelle des sommets.

En sortie, il renvoie leur position normalisée (implicitement) a travers un appel a gl_Position qui est envoyée au nuanceur de fragments.

- Le nuanseur de fragment: Il prend en entree la sortie du nuanceur de somment (position normalisée des fragments) et possiblement une entrée de couleur afin de definir la couleur de base de ces derniers. En sortie, il renvoie la couleur des fragments obtenue/résultante

Les sommets sont transformés en fragment a l'etape de tramage ou le primitives sont décomposées en petits morceaux que sont les fragments.

- La pipeline programmable permet d'adapter le code aux situations et de modier les elements pour varier le comportement.

dans une situatuion ou par exemple je suis dans un contexte de texture avec une pipeline standard, si je veux par exemple modifier le rendu pour avoir une texture qui bouge, si le pipeline n'est pas programmable cela ne me sera pas possible a part si je change de pipeline pour un autre standart qui permet les deplacement, cela n'est pas tres interessant

Commentaire :

Question 10

Terminé

Note de 6,00 sur 6,00

Marquer la question

Décrivez deux situations distinctes (excluant un miroir) où l'utilisation de stencil serait appropriée pour réaliser la modélisation d'une situation.

De manière brève, expliquer :

1. La situation choisie et pourquoi le stencil serait l'outil à préconiser
2. Comment le tampon de stencil se rempli (les étapes vues en cours)
3. De quelle façon les valeurs dans le tampon doivent changer en cas de réussite/échec du test de stencil [*glStencilOp(sfail, zfail, spass)*]

Il n'est pas attendu d'avoir les bonnes méthodes OpenGL avec les bons paramètres, mais une réponse en vos mots.

1- Prenons le cas ou nous souhaitons :

-nous souhaitons retirer une image d'une autre Le stencil nous permettrait de faire passer les copies de l'image sur differents emplacement desirés et le faire echouer l'image de devant

- on veut dessiner un rectangle en avant d'une autre figure disons un bonhomme . Nous souhaitons que les parties du bonhomme directement derriere le rectangle soit affichée quand meme a l,ecran, on pourrait dans cette zone faire passer le stencil du bonhomme ete chouer celui du rectangle

2- Techniquement, le stencil, applique un masque au stencil du fragment(valeur de référence stipulée) et on compare le resultat avec celui du stencil du tampon. De memoire, la zone concernée est remplie de '1' et si le test passe, il n'y aura que cette zone de représentée.

Aussi, il faut enable le test du stencil avant toutes choses avant d'appliquer le test/mask sinon le stencil ne changera pas.

recap:

- enable le test/mask

- appliquer le masque a la valeur specifiée puis la comparer a la valeur du tampon (Je ne me souviens plus exactement de la methode OpenGl qui le fait...)

- modifier le stencil en fonction du résultat obtenu grace a l'appel a glStencilOp()

3- en utilisant glStencilOp(sfail, zfail, spass) en fait:

* si le test de stencil echoue, l'opération sfail sera exécutée

* si le test de stencil reussi mais que celui de profondeur passe l'opération zfail est exécutée

* si les deux tests passent correctement (et celui de profondeur, et celui de stencil) alors spass est exécutée.

Les opérations peuvent etre des incrémentation, des replace etc