

# **INF2705 Infographie**

## **Spécification des requis du système**

### **Travail pratique 1**

#### ***Transformations et objets graphiques***

## Table des matières

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>                    | <b>2</b>  |
| 1.1      | But . . . . .                          | 2         |
| 1.2      | Portée . . . . .                       | 2         |
| 1.3      | Remise . . . . .                       | 2         |
| <b>2</b> | <b>Description globale</b>             | <b>3</b>  |
| 2.1      | But . . . . .                          | 3         |
| 2.2      | Travail demandé . . . . .              | 3         |
| <b>3</b> | <b>Exigences</b>                       | <b>7</b>  |
| 3.1      | Exigences fonctionnelles . . . . .     | 7         |
| 3.2      | Exigences non fonctionnelles . . . . . | 7         |
| <b>A</b> | <b>Liste des commandes</b>             | <b>8</b>  |
| <b>B</b> | <b>Apprentissage supplémentaire</b>    | <b>9</b>  |
| <b>C</b> | <b>Figures supplémentaires</b>         | <b>10</b> |

# 1 Introduction

Ce document décrit les exigences du TP1 « *Transformations et objets graphiques* » (Hiver 2022) du cours INF2705 Infographie.

## 1.1 But

Le but des travaux pratiques est de permettre à l'étudiant de directement appliquer les notions vues en classe.

## 1.2 Portée

Chaque travail pratique permet à l'étudiant d'aborder un sujet spécifique.

## 1.3 Remise

Faites la commande « `make remise` » ou exécutez/cliquez sur « `remise.bat` » afin de créer l'archive « **INF2705\_remise\_TPn.zip** » (ou .7z, .rar, .tar) que vous déposerez ensuite dans Moodle. (Moodle ajoute automatiquement vos matricules ou le numéro de votre groupe au nom du fichier remis.)

Ce fichier zip contient tout le code source du TP (`makefile`, `*.h`, `*.cpp`, `*.glsl`, `*.txt`).

## 2 Description globale

### 2.1 But

Le but de TP est de permettre à l'étudiant de mettre en pratique les fonctions de contrôle pour la modification des matrices d'OpenGL : `Rotate()`, `Translate()`, `Scale()`, `PushMatrix()` et `PopMatrix()`.

Ce travail pratique lui permettra aussi d'utiliser les fonctions liées aux *Vertex Buffer Objects (VBO)* : `glGenBuffers()`, `glBindBuffers()`, `glBufferData()` et `glDrawElements()`.

### 2.2 Travail demandé

#### Partie 1 : le drone

Les drones ont plusieurs domaines d'expertise tels que l'agriculture. Dans ce domaine, on retrouve plusieurs applications. On vous demande de réaliser un programme permettant d'afficher un drone afin de simuler une sonde d'exploration au-dessus d'un champs.<sup>1</sup> (figure 1).

Notre drone se déplace dans un espace limité et tourne sur lui-même pour regarder dans diverses directions (figure 2). Le drone a un corps cubique (légèrement aplati) et il possède quatre supports pour des hélices attachées en haut de son corps. Les pales du drone, situées sur un mat aux extrémités des supports, tournent en sens opposées l'une de l'autre (rotor contrarotatif).

Dans la deuxième partie de ce TP, vous ajouterez la possibilité pour ce drone de devenir LA théière bien connue en infographie. La figure 1 montre le corps du drone sous ses deux formes : cube ou théière.

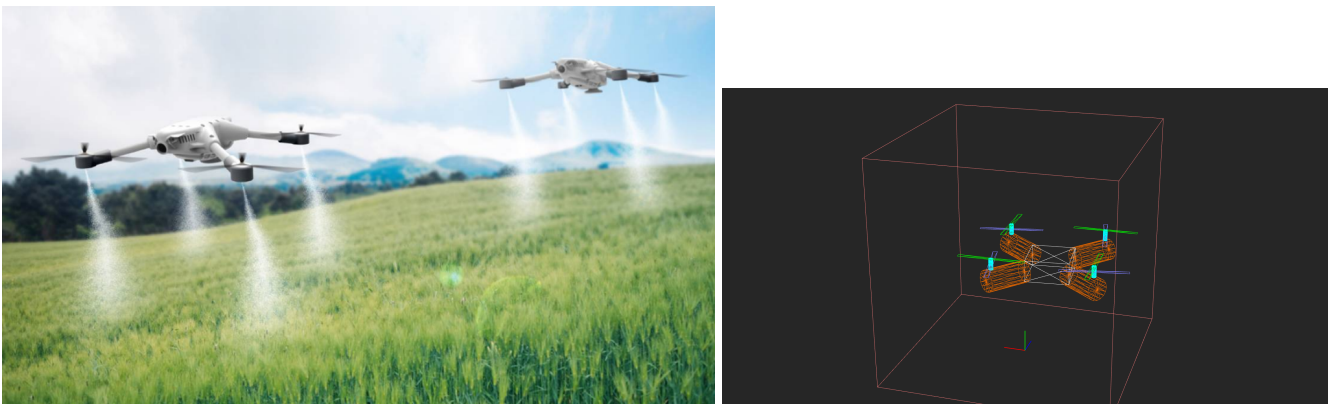


FIGURE 1 – Drone d'agriculture et notre modèle

1. <https://www.geoffreydromard.com/agricultural-drones-worth-getting-one/>

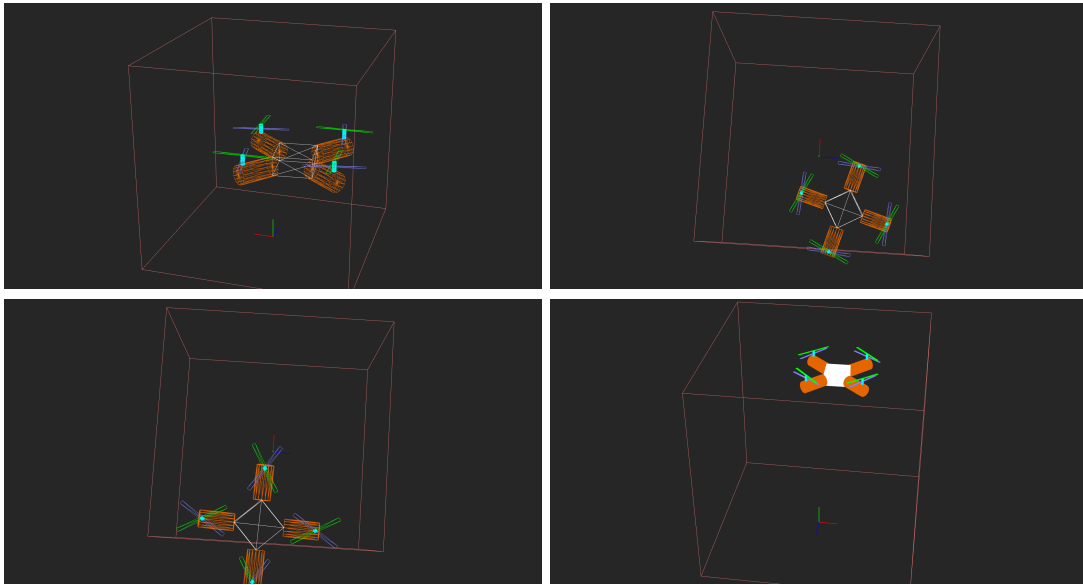


FIGURE 2 – Drone cube.

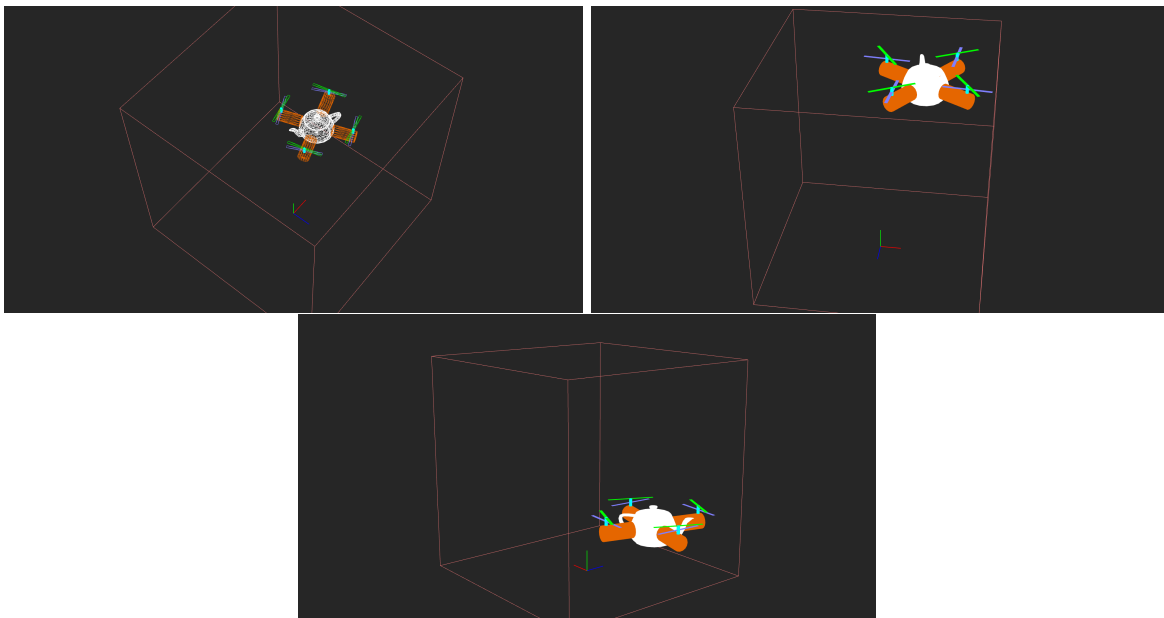


FIGURE 3 – Drone théière

Le drone est construit ainsi :

- est placé à la position courante « `position[]` »,
- est orienté selon l'angle « `angleCorps` » et
- est de taille : ( `taille`, `taille`, `taille` ),
- Son corps est un cube légèrement aplati en y (sa hauteur) : facteurs d'échelle = (2.0,1.0,2.0). Pour la partie 2, la taille de la théière sera simplement modulée par (0.5,0.5,0.5).
- Le bas du cube (corps) est à l'origine.
- Il y a 4 hélices, 4 supports à hélices et 4 rotors (entre les hélices et leurs mats).
- Aux 4 coins du corps, on retrouve les supports.
- Sur chacun de ces supports, il y a un rotor.
- Un rotor est composé d'un mat et de deux hélices.
- Le mat vertical est représenté par un cylindre vertical de taille (0.1,0.1,0.5), positionné à -0.25 de l'extrémité extérieur.
- Une pale est un quadrilatère de longueur « 3.0 » et de largeur « 0.2 ».
- La première pale est à hauteur de 0.25 au-dessus de la face supérieure du support et tourne dans le sens horaire lorsque vue du dessus.
- La seconde pale est à hauteur de 0.50 au-dessus de la face supérieure du support et tourne dans le sens antihoraire.
- L'angle de rotation des pales est donné par la variable « `anglePale` ».
- Le support est un cylindre de rayon 0.5 et de longueur 2.0.
- Le support est positionné au coin du corps à un angle de 45 degré.

Le cylindre, la sphère, les parallélépipèdes et les quadrilatères sont tracés par des appels aux fonctions fournies (sans modifier ces fonctions). Les valeurs des variables sont contrôlées interactivement.

La couleur de chaque partie du drone est donnée dans le tableau ci-dessous :

|         |        |                   |
|---------|--------|-------------------|
| Corps   | blanc  | ( 1.0, 1.0, 1.0 ) |
| Mat     | cyan   | ( 0.0, 1.0, 1.0 ) |
| Pale 1  | violet | ( 0.5, 0.5, 1.0 ) |
| Pale 2  | vert   | ( 0.0, 1.0, 0.0 ) |
| Support | marron | ( 0.9, 0.4, 0.0 ) |

## Partie 2 : utilisation de *Vertex Buffer Objects* (VBO)

On pourra représenter le corps du drone par un cube ou par la célèbre théière. La théière sera affichée en utilisant deux VBO (sommets et indices) créés avec les deux tableaux définis dans le fichier déjà inclus « `inf2705-theiere.h` ». Ces VBO doivent être définis une seule fois à l'initialisation et ensuite réutilisés à chaque affichage.

Pour vous inspirer, vous pouvez utiliser le triangle dans l'exemple vu au cours :

<https://gitlab.com/ozell/inf2705-exemples/-/tree/master/03-Base/src/main.cpp>,

<https://gitlab.com/ozell/inf2705-exemples/-/tree/master/03-Base/>.

## 3 Exigences

### 3.1 Exigences fonctionnelles

Partie 1 :

- E1. Toutes les parties du drone sont dessinées en utilisant les fonctions déjà définies : `afficherCylindre()`, `afficherQuad()` et `afficherCube()`. [2 pts]
- E2. Les fonctions `Rotate()`, `Translate()` et `Scale()` sont correctement utilisées pour les transformations géométriques nécessaires au dessin de chaque partie du drone. [1 pt]
- E3. Le drone est positionné selon `position`. La taille du modèle est donné par `taille`. L'orientation du drone suit bien `angleCorps`. [3 pts]
- E4. Les supports des rotors sont bien positionnés aux 4 coins du corps et chaque support est de rayon 0.5 et de longueur 2.0, et orientées de 45 degré. [2 pts]
- E5. Les mats du drone sont bien positionnées aux extrémités des supports au-dessus de ceux-ci et les pales tournent selon `anglePale`. [3 pts]
- E6. Les fonctions `PushMatrix()` et `PopMatrix()` sont correctement utilisées pour sauvegarder l'état des matrices. [2 pts]

Partie 2 :

- E7. Les fonctions `glGenBuffers()`, `glBindBuffers()`, `glBufferData()` et `glDrawElements()` sont correctement utilisées afin d'utiliser deux VBO (sommets et indices) pour afficher la théière. Les VBO sont définis une seule fois à l'initialisation et réutilisés à chaque affichage. [3 pts]
- E8. Le corps du drone peut être affichée en utilisant cette théière telle qu'illustrée à la figure 3. [2 pts]
- E9. (Le logiciel utilise correctement les touches listées à l'annexe A pour faire varier les divers paramètres.) (Correction négative).

### 3.2 Exigences non fonctionnelles

De façon générale, le code que vous ajouterez sera de bonne qualité. Évitez les énoncés superflus (qui montrent que vous ne comprenez pas bien ce que vous faites!), les commentaires erronés ou simplement absents, les mauvaises indentations, etc. [2 pts]

Pour la partie 1, des modifications sont principalement à faire dans les fonctions `afficher*()`. Pour la partie 2, des modifications sont principalement à faire dans les fonctions `initialiserGraphique()` et `afficherTheiere()`.



## ANNEXES

### A Liste des commandes

| <b>Touche</b> | <b>Description</b>  |
|---------------|---|
| q             | Quitter l'application                                     |
| x             | Activer/désactiver l'affichage des axes                   |
| i             | Réinitialiser le point de vue et l'hélicoptère            |
| g             | Permuter l'affichage en fil de fer ou plein               |
| c             | Permuter l'affichage des faces arrières                   |
| l             | Utiliser LookAt ou Translate+Rotate pour placer la caméra |
| m             | Choisir le modèle affiché : cube, théière                 |
| MOINS         | Reculer la caméra   |
| PLUS          | Avancer la caméra   |
| DROITE        | Déplacer l'hélicoptère vers +X                            |
| GAUCHE        | Déplacer l'hélicoptère vers -X                            |
| HAUT          | Déplacer l'hélicoptère vers +Y                            |
| BAS           | Déplacer l'hélicoptère vers -Y                            |
| PAGEPREC      | Déplacer l'hélicoptère vers +Z                            |
| PAGESUIV      | Déplacer l'hélicoptère vers -Z                            |
| f             | Diminuer la taille de la tête                             |
| r             | Augmenter la taille de la tête                            |
| VIRGULE       | Diminuer l'angle de rotation de l'hélicoptère             |
| POINT         | Augmenter l'angle de rotation de l'hélicoptère            |
| o             | Diminuer l'angle des pattes                               |
| p             | Augmenter l'angle des pattes                              |
| j             | Diminuer l'angle des pales                                |
| u             | Augmenter l'angle des pales                               |
| b             | Incrémenter la dimension de la boîte                      |
| h             | Décrémenter la dimension de la boîte                      |
| ESPACE        | Mettre en pause ou reprendre l'animation                  |
| s             | Sauvegarder une copie de la fenêtre dans un fichier       |
| BOUTON GAUCHE | Déplacer (modifier angles) la caméra                      |

## B Apprentissage supplémentaire

1. Quel est le nombre minimal de `PushMatrix()`/`PopMatrix()` à utiliser ? Pourquoi faut-il éviter d'en ajouter inutilement ?
2. Allonger les supports différemment de la taille du corps.
3. Ajouter des pales supplémentaires au drone.
4. Afin de mettre en pratique l'utilisation des transformations élémentaires, remplacer l'utilisation de la fonction `LookAt()`, en gardant le même point de vue, par une série de transformations utilisant des appels à `Translate()` et `Rotate()`.
5. Remplacer la fonction `LookAt()` qui positionne la caméra en orbite autour de la scène. Le `VecteurUp` devra être modifié en fonction de la position de l'observateur afin d'avoir l'impression que l'observateur est en orbite.
6. Si vous voulez que le drone se rende à une certaine position au-dessus du plan (p.e. pour y regarder quelque chose), comment allez-vous vous y prendre pour déterminer les valeurs appropriées des angles afin de positionner le corps à cet endroit ?  
(C'est une question qui relève de la « cinématique inverse ».)  
Y a-t-il plusieurs solutions possibles ?  
Y a-t-il des solutions qu'il faut éviter ?  
(Il faut aussi faire attention à ce que le drone ne sorte pas de sa terrain de jeu !)

## C Figures supplémentaires

Allez voir la thèière bien connue en infographie sur Internet :

[https://fr.wikipedia.org/wiki/Théière\\_de\\_l'Utah](https://fr.wikipedia.org/wiki/Théière_de_l'Utah)

[https://www.sjbaker.org/wiki/?title=The\\_History\\_of\\_The\\_Teapot.](https://www.sjbaker.org/wiki/?title=The_History_of_The_Teapot.)



FIGURE 4 – LA théière dans un écran de veille (Windows 95/98) <http://www.eeggs.com/items/2422.html>

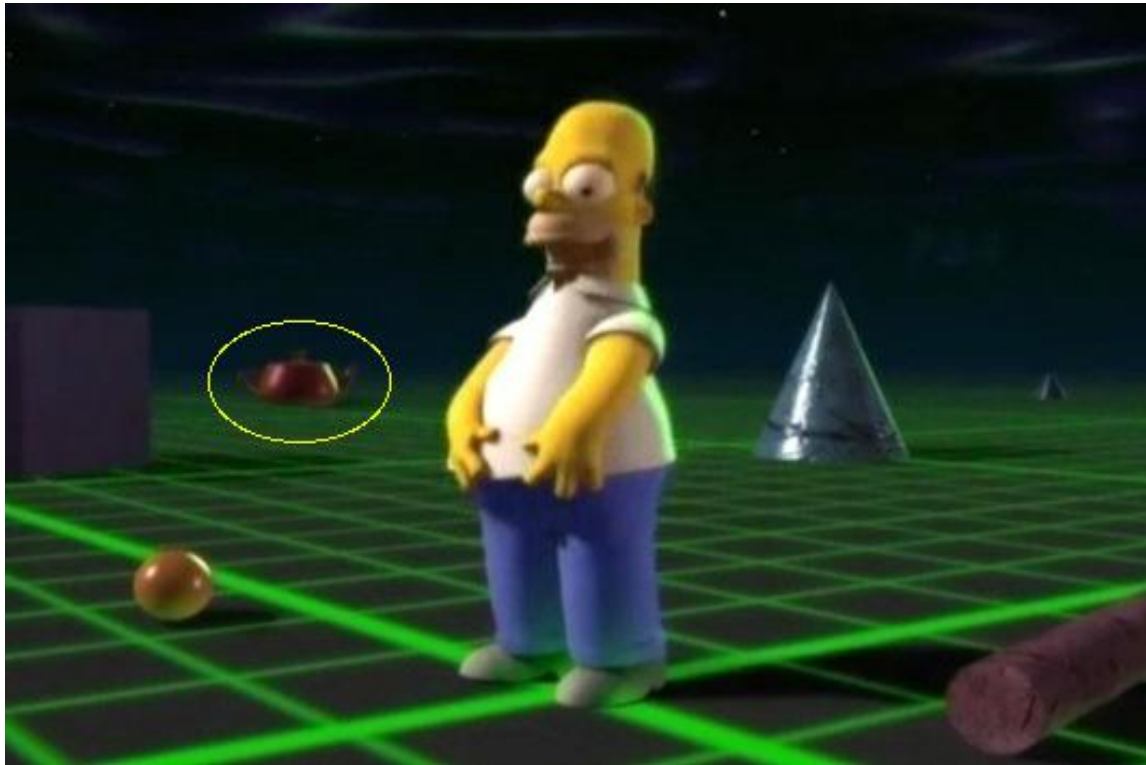


FIGURE 5 – Homer Simpson a vu LA théière dans *Treehouse of Horror VI*



FIGURE 6 – LA théière utilisée dans *Toy Story*



FIGURE 7 – LA théière, chez mon prédécesseur ! :)