Master 2 AVR
Université de Lorraine

Réalité augmentée

TP 1

ARToolKit est une bibliothèque écrite en C permettant d'effectuer un suivi optique (i.e. avec une caméra) basé marqueurs. Elle a été développée à la fin des années 90 par le Dr. Hirokazu Kato (Université d'Osaka au Japon) et le laboratoire Human Interface Technology (HIT Lab) de l'Université de Washington.

Compilation d'Artoolkit

1. Enregistrez le fichier glut.h dans un de vos répertoires

2. Allez dans le répertoire où vous souhaitez copier Artoolkit et lancez les commandes suivantes pour copier et compiler la librairie (et vérifier que ça marche) :

```
cp -r /opt/artoolkit5/ .
cd artoolkit5/
mkdir include/GL/
cp /chemin/vers/glut.h include/GL/
ln -s /usr/lib/x86_64-linux-gnu/libglut.so.3.9.0
./lib/libglut.so
./Configure
----------------------------------------
Standard compiler is gcc and g++ with libstdc++. Do you want to
use the Clang compiler with libc++ instead? (y or n)
Enter : n
Do you want to enable the Video4Linux2 capture module? (y or n)
Enter : y
Do you want to use it as the default input device? (y or n)
Enter : y
Do you want to enable the IEEE 1394 Digital Video Camera
capture module? (y or n)
Enter : n
Do you want to enable the GStreamer capture module? (y or n)
Enter : n
Enable the OpenSceneGraph renderer and examples? (y or n)
Enter : n
----------------------------------------
Make
cd bin
```

```
./simpleTest
```

→ Un cube doit apparaître sur le marqueur Hiro

```
cd ../examples/simple
emacs simpleTest.c&
Echap >
#if 1 => #if 0
Ctrl X Ctrl S
make
../../bin/simpleTest
```

→Une théière doit apparaître sur le marqueur Hiro

Exercice 1 – Une première application avec ARToolKit

ARToolKit utilise la bibliothèque GLUT (OpenGL Utility Toolkit) pour la gestion de la fenêtre (en OpenGL) et des interactions avec le clavier et la souris.

1. Modifiez le code de la fonction simpleTest.c pour afficher une théière (teapot en anglais…) à la place du cube.
2. Modifiez le programme pour qu'un appuie sur la touche « w » change l'affichage en mode fil de fer. Un nouvel appuie sur « w » remet l'affichage en mode surfaces.

Exercice 2 – Ordre des transformations

Un problème très souvent rencontré dans la manipulation d'objets virtuels 3D concerne l'ordre des transformations géométriques appliquées à un objet.

1. Nous allons commencer par essayer de comprendre où sont les axes dans le repère monde. Vous avez dû remarquer que le code de simpleTest.c commence par remettre la théière droite et posée (remontée) sur le marqueur (enlevez ces deux lignes pour vous en rendre compte). Modifiez le programme pour que chaque appui sur la touche « i » incrémente la rotation de la théière de 2 degrés autour de l'axe x, *après* que la théière ait été remise droite, mais *avant* qu'elle ne soit remontée. Faites de même pour l'axe y et l'axe z.

2. Modifiez le programme pour qu'un appuie sur la touche « a » fasse effectuer une rotation (non nulle) puis une translation (non nulle) quelconque à la théière. Un appuie sur la touche « z » effectuera les mêmes transformations mais dans l'ordre inverse. Un appuie sur la touche « e » ramènera l'objet à sa position d'origine. Le résultat est-il le même lorsque vous appuyez sur « a » et lorsque vous appuyer sur « z » ? Pourquoi ?

Exercice 3 : empilez et depilez des matrices…

En OpenGl, les transformations sont stockées sous forme de matrices, et sont mises dans une pile de matrices. Toutes les opérations matricielles que vous effectuez s'adressent à la matrice active, c'est-à-dire à la matrice au sommet de la pile. *glPushMatrix()* permet de recopiez la matrice active en haut de la pile et *glPopMatrix()* permet de détruire la matrice située au sommet de la pile. En d'autres termes, *glPushMatrix()* signifie « rappelle toi toi où tu es » et *glPopMatrix()* veut dire « retourne où tu étais ».

Dans un premier temps, associez deux objets virtuels différents à un marqueur. L'un est situé à 10cm à gauche du marqueur et l'autre à 10cm à droite du marqueur.
Ensuite un appuie sur la flèche droite rapprochera (i.e. une translation de x cm) les deux objets virtuels du centre du marqueur. Un appuie sur la flèche gauche les éloignera. Enfin « r » repositionne les objets virtuels à leur position d'origine. Vous utiliserez donc les deux fonctions *glPushMatrix()* et *glPopMatrix()* pour pouvoir effectuer des transformations différentes à vos deux objets virtuels…

Dans un second temps, créez un robot ayant les bras alignés le long de son corp. Un appui sur la barre d'espace devra lui faire lever les bras à 90 degrés.