Préambule

En OpenGL, comme d'ailleurs dans la plupart des logiciels ou librairies que vous utiliserez, vous aurez accès à des opérations de haut niveau afin de gérer les transformations et les projections. Mais encore faut-il être sûr de ce que vous allez faire. Un petit exercice afin de vous mettre dans le bain s'impose.

Exercice 1 : transformations

Les transformations disponibles sont : translation (*glTranslate*), rotation (*glRotate*), mise à l'échelle (*glScale*)

1. En modifiant la base (*base3d.c*) et à l'aide de ces seules transformations :
   a. afficher 4 théières (*glutWireTeapot*), alignées sur l'axe *[Oy)*, réparties de la plus grande à la plus petite ;
   b. faire tourner cet ensemble de théières autour de l'axe *[Oz)* ;
   c. faire tourner les théières autour d'elles-mêmes, en incrémentant la vitesse de rotation selon l'éloignement à *O* (image 1).

Note : l'ensemble de ces opérations doit prendre moins de 10 lignes.

2. créer, autour de chaque théière, une sphère, initialement sur le bec et tournant autour de la théière (désynchroniser des précédentes rotations), comme sur l'image 2.

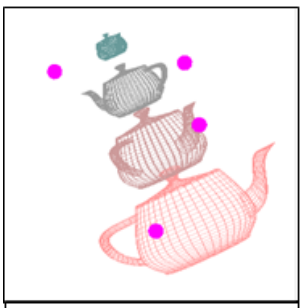


Figure 2

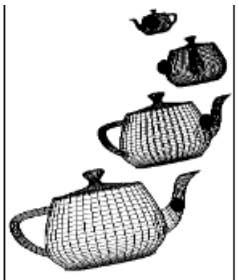


Figure 3

Petit problème : si pour faire tourner une sphère autour d'une théière, cela va entraîner l'empilement de la transformation pour les sphères et les théières suivante (image 3).

Solution : utiliser Push et PopMatrix() pour sauvegarder la matrice de transformation (de manière intelligente !).

Annexes 1 : Opérations diverses liées à la modélisation et à la visualisation

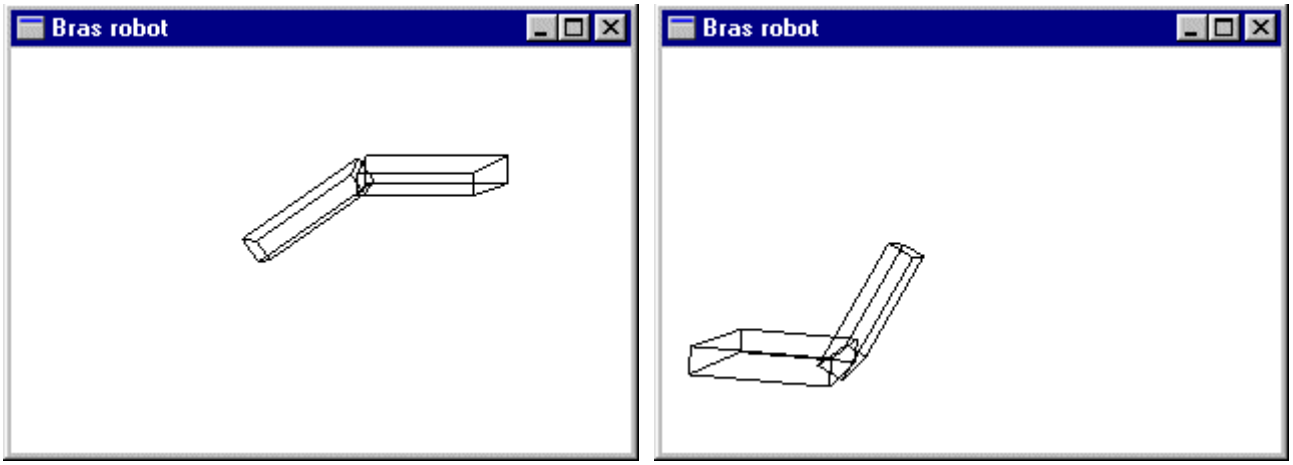
void glPushMatrix(void);

Empile la matrice courante dans la pile de matrices.
ATTENTION, il existe deux piles, une en mode GL_MODELVIEW et une en mode GL_PROJECTION.

void glPopMatrix(void);

Dépile la matrice en haut de pile du mode courant et remplace la matrice courante par celle-ci.

Annexe 2 : Exemple (extrait de http://raphaello.univ-fcomte.fr/ig/opengl/ExemplesGLUt/BrasRobot.htm)



Source

```
/*
 * Copyright (c) 1993-1997, Silicon Graphics, Inc.
 * ALL RIGHTS RESERVED
 * Permission to use, copy, modify, and distribute this software for
 * any purpose and without fee is hereby granted, provided that the abo
ve
 * copyright notice appear in all copies and that both the copyright no
tice
 * and this permission notice appear in supporting documentation, and t
hat
 * the name of Silicon Graphics, Inc. not be used in advertising
 * or publicity pertaining to distribution of the software without spec
ific,
 * written prior permission.
 *
 * THE MATERIAL EMBODIED ON THIS SOFTWARE IS PROVIDED TO YOU "AS-IS"
 * AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE,
 * INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR
 * FITNESS FOR A PARTICULAR PURPOSE.  IN NO EVENT SHALL SILICON
 * GRAPHICS, INC.  BE LIABLE TO YOU OR ANYONE ELSE FOR ANY DIRECT,
 * SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY
 * KIND, OR ANY DAMAGES WHATSOEVER, INCLUDING WITHOUT LIMITATION,
 * LOSS OF PROFIT, LOSS OF USE, SAVINGS OR REVENUE, OR THE CLAIMS OF
 * THIRD PARTIES, WHETHER OR NOT SILICON GRAPHICS, INC.  HAS BEEN
 * ADVISED OF THE POSSIBILITY OF SUCH LOSS, HOWEVER CAUSED AND ON
 * ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE
 * POSSESSION, USE OR PERFORMANCE OF THIS SOFTWARE.
 *
 * US Government Users Restricted Rights
 * Use, duplication, or disclosure by the Government is subject to
 * restrictions set forth in FAR 52.227.19(c)(2) or subparagraph
 * (c)(1)(ii) of the Rights in Technical Data and Computer Software
 * clause at DFARS 252.227-7013 and/or in similar or successor
 * clauses in the FAR or the DOD or NASA FAR Supplement.
 * Unpublished-- rights reserved under the copyright laws of the
 * United States.  Contractor/manufacturer is Silicon Graphics,
 * Inc., 2011 N.  Shoreline Blvd., Mountain View, CA 94039-7311.
 *
 * OpenGL(R) is a registered trademark of Silicon Graphics, Inc.
```

```
 */

/*
 * robot.c
 * This program shows how to composite modeling transformations
 * to draw translated and rotated hierarchical models.
 * Interaction:  pressing the s and e keys (shoulder and elbow)
 * alters the rotation of the robot arm.
 */

#include <stdio.h>
#include <stdlib.h>

#include <GL/glut.h>
#include <GL/gl.h>
#include <GL/glu.h>

#include "ModuleCouleurs.h"
#include "ModuleManipulateur.h"
#include "ModuleMenus.h"
#include "ModuleReshape.h"

static int shoulder = 0, elbow = 0;

void display(void) {
  glClear(GL_COLOR_BUFFER_BIT);
  glColor4fv(couleurNoir());
  glPushMatrix();
    manipulateurSouris();
    manipulateurClavier();
    glPushMatrix();
      glTranslatef(-1.0,0.0,0.0);
      glRotatef((GLfloat) shoulder,0.0,0.0,1.0);
      glTranslatef(1.0,0.0,0.0);
      glPushMatrix();
        glScalef(2.0F,0.4F,1.0F);
        glutWireCube(1.0);
      glPopMatrix();
      glTranslatef(1.0,0.0,0.0);
      glRotatef((GLfloat) elbow,0.0,0.0,1.0);
      glTranslatef(1.0,0.0,0.0);
      glPushMatrix();
        glScalef(2.0F,0.4F,1.0F);
        glutWireCube(1.0);
      glPopMatrix();
    glPopMatrix();
  glPopMatrix();
  glFlush();
  glutSwapBuffers();
}

void myinit(void) {
  glClearColor(1.0,1.0,1.0,1.0);
}

void special(int k, int x, int y) {
  switch (k) {
```

```
    case GLUT_KEY_UP    : elbow = (elbow + 5) % 360;
                          glutPostRedisplay();
                          break;
    case GLUT_KEY_DOWN  : elbow = (elbow - 5) % 360;
                          glutPostRedisplay();
                          break;
    case GLUT_KEY_LEFT  : shoulder = (shoulder + 5) % 360;
                          glutPostRedisplay();
                          break;
    case GLUT_KEY_RIGHT : shoulder = (shoulder - 5) % 360;
                          glutPostRedisplay();
                          break; }
}

int main(int argc,char **argv) {
  glutInit(&argc,argv);
  glutInitDisplayMode(GLUT_RGBA|GLUT_DOUBLE);
  glutInitWindowSize(300,300);
  glutInitWindowPosition(50,50);
  glutCreateWindow("Bras robot");
  myinit();
  creationMenuBasique();
  setParametresPerspectiveBasique(65.0F,1.0F,1.0F,20.0F,0.0F,0.0F,-
5.0F);
  setManipulateurDistance(5.0F);
  glutReshapeFunc(reshapePerspectiveBasique);
  glutKeyboardFunc(keyBasique);
  glutSpecialFunc(special);
  glutMotionFunc(motionBasique);
  glutMouseFunc(sourisBasique);
  glutDisplayFunc(display);
  glutMainLoop();
  return(0);
}
```