EI3-Semestre 1/IHM-RV/TD n°1

Objectifs des travaux pratiques et outils considérés.

L'objectif est découvrir, par la programmation, les interactions homme machine (événements clavier, souris et menus, avec la librairie GLUT), l'affichage 2D et l'affichage 3D (avec la librairie OpenGL). OpenGL est la librairie de référence pour la 3D, librairie sur laquelle sont basés la plupart de moteurs 3D avancés (i.e. fonctions avancées). Bien que ces libraires soient nativement écrites en langage C, nous choisissons de travailler avec la surcouche Python : l'appel d'une fonction en Python se traduit par l'appel implicite à la fonction C sous-jacente portant le même nom. Ce choix est motivé par la simplicité de Python, permettant ainsi de se concentrer sur la découverte des notions interactions/2D/3D, en évitant la (relative) complexité du langage C.

Remarque : Les programmes Python sont en général plus « lent » que les programmes écrits en C : Python permet de prototyper un programme qui pourrait, pour une version aboutie, être « traduite » en C.

Documentation pratiques : les fonctions mentionnées sur le site « PyOpenGL » sont détaillées les sites « GLUT » et « OpenGL » (en langages C)

- Python : outil de développement Python recommandé: IDLE ; documentation « langage » fournie + documentation en ligne
- Documentation GLUT/OpenGL (Python): *http://pyopengl.sourceforge.net/documentation/manual-3.0/index.html*
- Documentation GLUT (en C) : *https://www.opengl.org/resources/libraries/glut/spec3/spec3.html*
- Documentation OpenGL (C) : https://www.opengl.org/sdk/docs/man2/

Consultation de la documentation GLUT: exemple de la fonction « glutMouseFunc » fournie par le site PyOpenGL.

Documentation en langage Python (site PyOpengl) → renvoi sur la documentation de référence OpenGL (langage C)

#Function est le « handler » passé en paramètre à glutMouseFunction, dont les paramètres sont button, state, x,y → site OpenGL
glutMouseFunc(fonction)
Specify handler for GLUT 'Mouse' events
def handler((int) button, (int) state, (int) x, (int) y): return None

Documentation en langage C (site OpenGL)

Usage
*void glutMouseFunc(void (*func)(int button, int state, int x, int y));*
func : The new mouse callback function.
Description
glutMouseFunc sets the mouse callback for the current window. When a user presses and releases mouse buttons in the window, each press and each release generates a mouse callback. The button parameter is one of GLUT_LEFT_BUTTON, GLUT_MIDDLE_BUTTON, or GLUT_RIGHT_BUTTON. For systems with only two mouse buttons, it may not be possible to generate GLUT_MIDDLE_BUTTON callback. For systems with a single mouse button, it may be possible to generate only a GLUT_LEFT_BUTTON callback. The state parameter is either GLUT_UP or GLUT_DOWN indicating whether the callback was due to a release or press respectively. The x and y callback parameters indicate the window relative coordinates when the mouse button state changed. If a GLUT_DOWN callback for a specific button is triggered, the program can assume a GLUT_UP callback for the same button will be generated (assuming the window still has a mouse callback registered) when the mouse button is released even if the mouse has moved outside the window. If a menu is attached to a button for a window, mouse callbacks will not be generated for that button. During a mouse callback, glutGetModifiers may be called to determine the state of modifier keys when the mouse event generating the callback occurred. Passing NULL to glutMouseFunc disables the generation of mouse callbacks.

Consultation de la documentation OpenGL: exemple de la fonction « glVertex ».

Documentation en langage Python (site PyOpengl) → renvoi sur la documentation de référence OpenGL (langage C)

Signature
*glVertex( )-> glVertex(* args) Choose glVertexX based on number of args*
[...]
glVertex2i(GLint (x) , GLint (y))-> void
def glVertex2i(x , y)
glVertex3f(GLfloat (x) , GLfloat (y) , GLfloat (z))-> void
def glVertex3f(x , y , z)
[...]
***Parameters** x, y, z, w*
Specify x , y , z , and w coordinates of a vertex. Not all parameters are present in all forms of the command.

Documentation en langage C (site OpenGL)

Specification
[...]
void glVertex2i(GLint x, GLint y);
void glVertex3f(GLfloat x, GLfloat y, GLfloat Z);
[...]
Parameters :
Specifies a pointer to an array of two, three, or four elements. The elements of a two-element array are x and y; of a three-element array, x, y, and z; and of a four-element array, x, y, z, and w.
Description
glVertex commands are used within glBegin/glEnd pairs to specify point, line, and polygon vertices. The current color, normal, texture coordinates, and fog coordinate are associated with the vertex when glVertex is called. When only x and y are specified, z defaults to 0 and w defaults to 1. When x, y, and z are specified, w defaults to 1.

Jean-Baptiste Fasquel

Objectif: Manipulation d'une interface homme machine simple en utilisant de la librairie GLUT.

Eléments fournis :

- Vidéo du programme à réaliser (lecture avec VLC par exemple)
- Code source du programme de départ et quelques tutoriels: on utilisera la documentation en ligne pour réaliser le travail

Le code source du programme de départ fournit plusieurs éléments :

- Initialisation de l'application
- Affichage d'un texte et d'un repère 2D
- Exemple d'un menu et sous-menu (bouton gauche souris)
- Exemple de gestion des événements (voir rappel du principe en fin de document):
  - Affichage associé à glutDisplayFunc()
  - Redimensionnement de la fenêtre associé à glutReshapeFunc()
  - Traitement d'un événement souris modifiant le texte affiché avec glutMouseFunc(): cet exemple fournit notamment les coordonnées sélectionnées dans le repère de la fenêtre (l'origine étant située en haut à gauche: (0,x) vers la droite et (0,y) vers le bas). Ces coordonnées sont distinctes de celles utilisées pour positionner le texte (repère (0,x,y) affiché). Cette distinction entre les systèmes de coordonnées est illustré par l'illustration 1.
- Un certain nombre de variables globales configurant l'affichage: position et taille du texte, contenu de la chaîne de caractères, zone affichée (carré d'arête 2), taille de la fenêtre.

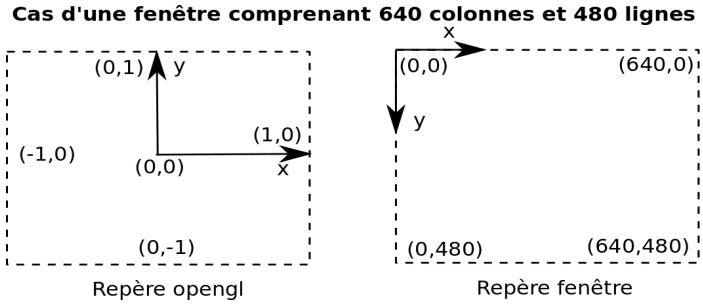

Illustration 1: Systèmes de coordonnées considérés.

L'objectif est de compléter ce programme pour reproduire l'exécutable fourni, en ajoutant et/ou adaptant des fonctions utilisant et/ou modifiant les variables globales proposées.

1) Gestion de l'affichage de la touche pressée au clavier (caractère ASCII). Pour déclarer la fonction gérant cet aspect, on utilisera glutKeyboardFunc() et on créera la fonction appropriée.

2) Gestion du déplacement (flèches) et de l'agrandissement/réduction (« zoom » - touches PAGE_UP et PAGE_DOWN) du texte. Pour déclarer la fonction gérant cet aspect, on utilisera glutSpecialFunc() et on créera la fonction appropriée.

3) Faire le menu « reset », et l'opération de réinitialisation.

4) Gérer l'affichage du texte à l'endroit sélectionné avec la souris (adapter MouseFunc). Les coordonnées sélectionnées dans la fenêtre sont passées en paramètre (repère fenêtre – voir illustration 1). Ces coordonnées sont différentes des coordonnées à assigner au texte (à exprimer dans le repère opengl - voir illustration 1).

5) Gérer le démarrage/arrêt de la fonction idleFunc() à partir du sous-menu (item « tâche de fond » dans l'application finale).

   - Pour le démarrage de la tâche de fond, on invoquera « glutIdleFunc(maFonctionTacheDeFond) ». Votre fonction « maFonctionTacheDeFond » sera invoquée indéfiniment (voir documentation de « glutIdleFunc »).
   - Pour l'arrêt de la tâche de fond, on invoquera « glutIdleFunc(NULL) ». Votre fonction sera ainsi remplacée par l'élément « NULL », et s'exécutera donc plus : la tâche de fond est alors arrêtée.

6) Gérer l'animation, déclenchée à partir du menu: il s'agit de compléter la fonction *Animation* et de l'invoquer à partir du menu approprié (item « tâche de fond » dans l'application finale).

Jean-Baptiste Fasquel

Gestion des événements avec GLUT: la librairie GLUT (il existe beaucoup d'autres librairies) permet de cacher à l'utilisateur l'interception d'événements de type « clic souris », « touche du clavier pressée », « fenêtre redimensionnée », … etc... Afin de personnaliser le comportement d'un programme par rapport à ces événements, on peut spécifier la fonction (« callback ») à exécuter lorsque l'un de ces événements se produit : cette fonction est passée comme argument à une fonction de la librairie GLUT. La signature de la fonction (e.g. nombre et type des paramètres) dépend du type d'événement considéré.

Structure typique d'un programme permettant de personnaliser le traitement des événements en utilisant la librairie GLUT : cas particulier des événements de redimensionnement de la fenêtre et des événements « souris ».

```
####################
#VARIABLE GLOBALE
####################
Lettre="L"
####################
#FONCTIONS « CALLBACKS »
####################
def Display():  #Gestion du rafraichissement
   glClear(GL_COLOR_BUFFER_BIT );     #Vide le buffer de couleur
   glMatrixMode(GL_MODELVIEW);     #On dessine l'objet
   glLoadIdentity() ;glPushMatrix() ;glScalef(0.005,0.005,0.005)
   glutStrokeCharacter(GLUT_STROKE_ROMAN,ord(Lettre)) #Dessine la « Lettre » (variable globale)
   glPopMatrix()
   glutSwapBuffers(); #Execute toutes les commandes en attente (i.e. affichage des caracteres), et permute les buffers ecriture et affichage

#Déclaration de la fonction associée au choix d'un élément du menu
#'selection' renseigne le numéro de l'élément sélectionné dans le menu
def Menu(selection): #Gestion du menu contextuel
   global Lettre #Pour préciser que la variable globale « Lettre » sera éditée (puis lue dans la fonction « Display »)
   if selection == 0: Lettre="A"
   elif selection == 1: Lettre="B"
   glutPostRedisplay() #Déclenche le rafraîchissement (par appel « dès que possible » à la fonction associée à glutDisplayFunc (« Display »)
   return 0

#Déclaration de la fonction associée à une interaction avec la souris
#le bouton de la souris concerné par l'événement est renseigné dans la variable 'button', son état (i.e. enfoncé, relâché) est renseigné par la variable
#'state', tandis que xscreen et yscreen donnent les coordonnées (dans le repère de la fenêtre) de la souris lors de l'événement
def Mouse(button, state, xscreen, yscreen):
   global Lettre
   if (button == GLUT_LEFT_BUTTON ) & (state == GLUT_DOWN):
            Lettre="B"
            glutPostRedisplay(); #Déclenche le rafraîchissement (par appel « dès que possible » à la fonction associée à glutDisplayFunc (« Display »)
####################
#MAIN : INITIALISATION & « AFFECTATION DES CALLBACKS »
####################
#Initialisation de la fenetre
glutInit(sys.argv) #Initialisation de GLUT par rapport au systeme graphique (e.g. serveur graphique)
glutCreateWindow('Window') #Creation fenetre principale avec titre

#Callbacks : chaque « callback » doit respecter une signature (i.e. paramètres) documentée
glutDisplayFunc(Display) #Affichage: invocation directe ou indirecte (glutPostRedisplay() requete d'appel "des que possible")
glutMouseFunc(Mouse) #Souris : la documentation de glutMouseFunc précise que la fonction (paramètre) « Mouse » doit avoir 4 paramètres
glutCreateMenu(Menu) #Menu

glutAddMenuEntry("Print A",0) #Menu item
glutAddMenuEntry("Print B",1) #Menu item
glutAttachMenu(GLUT_RIGHT_BUTTON) #Evenement d'affichage du menu

#Demarrage de l'interface graphique avec « attente » des événements (e.g. clics 'souris')
glutMainLoop();
```