Lab 3.1: Mouse Interaction & Write to Text File

In this Lab we shall develop a facility whereby we can interact with the position of the mouse pointer and either draw the position or use it to locate objects for later processing. To illustrate this process we will develop software to simulate the mouse behaving as a pen. The (x, y) coordinates of the generated points are written to a text file. The GLUT kit provides callback functions to handle interrupts from the mouse via glutMouseFunc and glutMotionFunc.

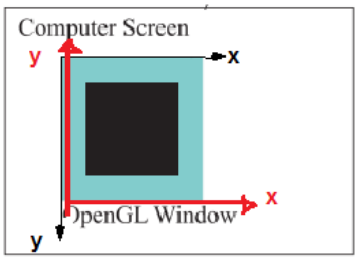
glutMouseFunc: registers callback handler for mouse click.

```
void glutMouseFunc (int button, int state, int x, int y)
    // (x, y) is the mouse-click location.
    // button: GLUT_LEFT_BUTTON, GLUT_RIGHT_BUTTON, GLUT_MIDDLE_BUTTON
    // state: GLUT_UP, GLUT_DOWN
```

glutMotionFunc: registers callback handler for mouse motion (when the mouse is clicked and moved).

```
void glutMotionFunc(void (*func)(int x, int y)
    // where (x, y) is the mouse location in Window's coordinates
```

Note: In DrawSquare, there is a need to convert y from screen coordinates convention to OpenGL coordinates. (y = wh – y);



black axis screen coordinate in pixels

red axis Open GL window.

The two Y axis are opposite to each other

Example Code: Mouse behaving as a pen

```
//Mouse_Pen.cpp

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <iostream>
#include <fstream> //Mouselnp.cpp
#include <windows.h>
#include <GL/glut.h>

GLsizei wh   = 250;
GLsizei ww   = 250;
GLfloat size = 3.0;  // half side of square
using namespace std; using std::ofstream;
ofstream drawfile ("C:/???? /out.txt");)// insert here the path of the file to write to
```

```
void Display    ( void );
void DrawSquare ( int x, int y );
void Mouse      ( int btn, int state, int x, int y );
void MyInit     ( );
void Reshape    ( int w, int h );

void Display ( void ) {

   glClear ( GL_COLOR_BUFFER_BIT );
            }

void DrawSquare ( int x, int y ) {
            //convert y from Windows convention to OpenGL
   y = wh - y;

   //set up random colour
   glColor3ub (       (GLubyte)rand( ) % 256,
                                          (GLubyte)rand( ) % 256,
              (GLubyte)rand( ) % 256 );
  drawfile << x << " " << y << endl;  // store colour as well?
   glBegin ( GL_POLYGON );
            glVertex2f ( x + size, y + size );
      glVertex2f ( x - size, y + size );
      glVertex2f ( x - size, y - size );
      glVertex2f ( x + size, y - size );
   glEnd ( );
   glFlush ( );
            }

void Mouse ( int btn, int state, int x, int y ) {
            if ( GLUT_LEFT_BUTTON == btn && GLUT_DOWN == state )
            DrawSquare ( x, y );
   if ( GLUT_RIGHT_BUTTON == btn && GLUT_DOWN == state )
            exit ( 0 );
            }

void MyInit ( void ) {
            glViewport ( 0, 0, ww, wh );
   // make clipping area match size of window
   glMatrixMode   ( GL_PROJECTION );
   glLoadIdentity ( );
   gluOrtho2D     ( 0.0, (GLdouble)ww, 0.0, (GLdouble)wh );
   glMatrixMode   ( GL_MODELVIEW );
   glClearColor   ( 0.0, 0.0, 0.0, 1.0 );
   glClear        ( GL_COLOR_BUFFER_BIT );
            } // end of function MyInit

void Reshape ( int w, int h ) {
   glMatrixMode   ( GL_MODELVIEW );
   glLoadIdentity ( );
   glViewport     ( 0, 0, w, h );
   glClear        ( GL_COLOR_BUFFER_BIT );
   glutPostRedisplay ( );
   // update globals
   ww = w;
   wh = h;
            }

int main(int argc, char **argv) {
```

```
    glutInit            ( &argc, argv );
    glutInitDisplayMode    ( GLUT_SINGLE | GLUT_RGB );
    glutInitWindowSize     ( ww, wh );
    glutInitWindowPosition ( 350, 50 );
    glutCreateWindow       ( "Mouse Draw" );
    MyInit ( );
    glutDisplayFunc ( Display );
    glutMouseFunc   ( Mouse );
    glutMotionFunc  ( DrawSquare );
    glutReshapeFunc ( Reshape );
    glutMainLoop ( );
            return 0;
}
```

Screen Shot of: Mouse Draw

The word "Hello" has been drawn by moving the mouse on the screen while pressing the left button + Writing generated coordinates to output file.

Lab 3.2: Reading From File & Draw Polygonal Snoopy

```
#include <stdio.h>  //line_poly_file1.cpp
#include <stdlib.h>
#include <string.h>
#include <iostream>
#include <fstream>
#include <windows.h>
#include <GL/glut.h>

GLsizei wh   = 250;
GLsizei ww   = 250;

// function prototypes
void DisplaySnoopy ( void );
void MyInit ( );
// end of function prototypes

void DisplaySnoopy ( void ) {
char line_fill;
GLfloat xj, yj;
int j, no_of_pts;
using namespace std;
using std::ifstream;using std::cerr;using std::endl;
fstream inStream;
inStream.open ("C:/????/snoopy3.txt", ios ::in);// insert here the path of the file to read from
```

```
if (!inStream) { cout << "File would not open\n";
                                                return;
                                        }
glClear ( GL_COLOR_BUFFER_BIT );
glScalef(5.0, 5.0, 1.0);
do {
            inStream >> no_of_pts >> line_fill;
            if(line_fill == 'l') glBegin (GL_LINE_STRIP);
            else glBegin (GL_POLYGON);
   for (j=1; j<=no_of_pts; j++) {
                        inStream >> xj >> yj;
                        glVertex2f (xj,yj);
   }
   glEnd();
} while ((xj = inStream.get()) != EOF);
glFlush ();
}

void MyInit ( void ) {
   glClearColor   ( 1.0, 1.0, 0.0, 0.0 ) ; //yellow background
   glColor3f(0.0f, 0.0f,1.0f);            // blue drawing colour
   glMatrixMode   ( GL_MODELVIEW ) ;
   glLoadIdentity ( ) ;
   gluOrtho2D     ( 0.0, (GLdouble)ww, 0.0, (GLdouble)wh ) ;
}

int main(int argc, char **argv) {
   glutInit             ( &argc, argv );
   glutInitDisplayMode    ( GLUT_SINGLE | GLUT_RGB );
   glutInitWindowSize     ( ww, wh );      // window size
   glutInitWindowPosition ( 50, 50 );      // window position on screen
   glutCreateWindow       ( "Sleeping Snoopy" );

   MyInit ( );

   glutDisplayFunc ( DisplaySnoopy );
   glutMainLoop ( );
   return(0);
}
```

Screen Shot of: Poly_Snoopy