#include<Windows.h>

#include<GL/glut.h>

#include<stdlib.h>

#include<math.h>

#include<conio.h>

#include<stdio.h>

#include<iostream>

#include<iomanip>

#include<gl/glut.h>

using namespace std;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*GLOBAL VALUES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

float theta = 0.0; //global angular value for rotation

float scale1 = 1.0; //global scaling value

float dx = 7.0, dy = -3.0;

int frame = 1;

int x=0;

void init(void); //This is a function to initialize the window clear color

void RenderScene(void); //This a function to draw polyman in an opened window

void loadicon(float[], float[], float[], float[], float[], float[],float[],float[], float[], float[],float[]); //Load the polyman icon

void drawicon(float[], float[], float[], float[], float[], float[], float[], float[], float[], float[],float[]); //Draw the icon the two first float for the square and the others for the line

void settrans(void); //Sets the transformation matrix for desired scale, rotation, new pos

void myidle(void);

void SetupRC(void); //Sets up the clear color

void TimerFunction(int); //This call back function is called each 30ms and changes the location, scale and rotation

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int main(int argc, char\*\* argv)

{

//Set up window title

char header[] = "Polyman By Chris Stewart";

glutInit(&argc, argv);

//Set up the display mode with two buffers and RGB colors

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

//Initialize window size and position

glutInitWindowSize(560, 440);

glutInitWindowPosition(140, 20);

//Initialize background color of the window

SetupRC();

//Open and label window

glutCreateWindow(header);

glutDisplayFunc(RenderScene);

glutTimerFunc(30, TimerFunction, 1); //Call the TimerFunction each 30s

//Now draw the scene

glutMainLoop();

return 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*RenderScene Function\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void RenderScene(void)

{

float xdel = 0.25;

float px[7], py[7], plx[4], ply[4], pl2x[4], pl2y[4], pl3x[3], pl3y[3], pl4x[4], pl4y[4], eye[2];// These variables hold the pattern for the icon square plus line

//clear the window with the current background color

cout << "in RenderScene" << endl;

//set the current drawing color to white

glColor3f(1.0, 1.0, 1.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

//set the viewport to the window dimensions

glViewport(0, 0, 560, 440);

//Establish the clipping volumn in user units, first clear all the translation matrices

glOrtho(-7.0, 7.0, -7.0, 7.0, 1.0, -1.0);

loadicon(px, py, plx, ply, pl2x, pl2y,pl3x,pl3y,pl4x,pl4y, eye);

//draw the icon untransformed

settrans();

//clear the window with the background color

glClear(GL\_COLOR\_BUFFER\_BIT);

//set the current drawing color to white

glColor3f(1.0, 1.0, 1.0);

//now draw the figure

drawicon(px, py, plx, ply, pl2x, pl2y,pl3x,pl3y,pl4x,pl4y,eye);

glEnd();

glutSwapBuffers();

return;

}//end of render scene

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LOAD ICON FUNCTION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void loadicon(float px[], float py[], float plx[], float ply[], float pl2x[], float pl2y[], float pl3x[], float pl3y[], float pl4x[], float pl4y[],float eye[]) //Loads the polyman

{

//Set the coordinates of the body

px[0] = -1.125; py[0] = 0.0;

px[1] = -0.625; py[1] = 0.75;

px[2] = 0.625; py[2] = 0.75;

px[3] = 1.125; py[3] = -0;

px[4] = 0.625; py[4] = -0.75;

px[5] = -0.625; py[5] = -0.75;

px[6] = -1.125; py[6] = 0.0;

//set the right foot

plx[0] = -0.25; ply[0] = -0.5;

plx[1] =- 0.25; ply[1] = -1.0;

plx[2] = -0.50; ply[2] = -1.0;

plx[3] = -0.25; ply[3] = -1.0;

//set the left foot

pl2x[0] = 0.25; pl2y[0] = -0.5;

pl2x[1] = 0.25; pl2y[1] = -1.0;

pl2x[2] = 0.0; pl2y[2] = -1.0;

pl2x[3] = 0.25; pl2y[3] = -1.0;

//set the Closed mouth

pl3x[0] = -0.375; pl3y[0] = -0.0;

pl3x[1] = -0.875; pl3y[1] = -0.5;

pl3x[2] = -0.875; pl3y[2] = -0.5;

//set the Open mouth

pl4x[0] = -1.125; pl4y[0] = 0.0;

pl4x[1] = -0.375; pl4y[1] = 0.0;

pl4x[2] = -0.625; pl4y[2] = -0.75;

pl4x[3] = -1.125; pl4y[3] = 0.0;

//set the eye

eye[0] = -0.5; eye[1] = 0.5;

return;

} //end of loadicon

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FUNCTION DRAWICON\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void drawicon(float pxp[], float pyp[], float plxp[], float plyp[], float pl2xp[], float pl2yp[], float pl3xp[], float pl3yp[], float pl4xp[], float pl4yp[],float eye[])

{

//draw the square icon at the transformed position

int i;

cout << "in drawicon" << endl;

glBegin(GL\_LINE\_STRIP);

//move to first point in the icon

glVertex2f(pxp[0], pyp[0]);

//now draw the rest of the box

for (i = 1; i <= 6; i++)

{

glVertex2f(pxp[i], pyp[i]);

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

//now draw the line

glBegin(GL\_LINES);

glVertex2f(plxp[0], plyp[0]);

for (i = 1; i <= 3; i++)

{

glVertex2f(plxp[i], plyp[i]);

}//glVertex2f(plxp[2],plyp[2]);

glEnd();

glBegin(GL\_LINES);

glVertex2f(pl2xp[0], pl2yp[0]);

for (i = 1; i <= 3; i++)

{

glVertex2f(pl2xp[i], pl2yp[i]);

}

glEnd();

//set the shading color to purple

glColor3f(1.0, 0.0, 1.0);

glShadeModel(GL\_FLAT);

//redraw the polygon

glBegin(GL\_POLYGON);

//Firts point is where the line intersects the top of the square

glVertex2f(pxp[0], pyp[0]);

//rigth corner upper

glVertex2f(pxp[1], pyp[1]);

//right corner lower

glVertex2f(pxp[2], pyp[2]);

//left intersect

glVertex2f(pxp[3], pyp[3]);

glVertex2f(pxp[4], pyp[4]);

glVertex2f(pxp[5], pyp[5]);

glVertex2f(pxp[6], pyp[6]);

glEnd();

//Draw Eye

glBegin(GL\_POINTS);

glColor3f(0.0, 0.0, 0.0);

glVertex2f(eye[0], eye[1]);

glEnd();

//Draw mouth

glColor3f(1.0, 1.0, 1.0);

if (dx == 0) {//Open mouth

glColor3f(0.0, 0.0, 0.0);

glShadeModel(GL\_FLAT);

//draw the polygon

glBegin(GL\_POLYGON);

glVertex2f(pl4xp[0], pl4yp[0]);

glVertex2f(pl4xp[2], pl4yp[2]);

glVertex2f(pl4xp[1], pl4yp[1]);

glVertex2f(pl4xp[3], pl4yp[3]);

}

else {//ClosedMouth

glBegin(GL\_LINES);

glVertex2f(pl3xp[1], pl3yp[1]);

glVertex2f(pl3xp[0], pl3yp[0]);

glVertex2f(pl3xp[2], pl3yp[2]);

}

return;

} //end of draw icon

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FUNCTION SETTRANS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* function settrans \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void settrans(void)

/\*Sets the MODELVIEW MATRIX for the square. Note that the calls are done backqards

that is if we want to rotate and move the pattern, call glTranslate first and then glRotate \*/

{

cout << "in settrans" << endl;

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(dx, dy, 0.0);

glRotatef(theta, 0.0, 0.0, 1.0);// note that the angle theta is in degrees, not radians

return;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Function SetupRC\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Setup the rendering state

void SetupRC(void)

{// this function sets the clear color of an open window and clears the open window

// Set clear color to blue

glClearColor(0.0, 0.0, 1.0, 1.0);

return;

}//end of SetupRC

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Functioner Timer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void TimerFunction(int value)

//this call back function is call each 30 ms and changes the location,scale and rotation

// of the square.

{

switch (frame)

{case 1: //frame 1 polyman starts at right (7, -3) and walks to the middle right (3.5,-3)

//Polyman parameters

dx -= 0.15;

theta = theta-5;

if (dx <= 0.0) {

dx = 0.0;

frame = 3;

}

else {

frame = 2;

}

break;

case 2: //frame 1 polyman starts at right (7, -3) and walks to the middle right (3.5,-3)

//Polyman parameters

theta = theta +5;

if (dx <= 0.0) {

dx = 0.0;

frame = 3;

}

else {

frame = 1;

}

//we key on Polyman's position to change the frame

break;

case 3:// frame 2 polyman jumps to y=5

dy += 0.2;

if (dy > 5.0)

{dy = 5.0;

frame = 4;

}

break;

case 4:// frame 3 Polyman rotates at x=3.5,y=5.0

theta += 5.0;

if (theta >= 360.0)

{frame = 5;

theta = 0.0;

}

break;

case 5: // frame 4 Polyman moves down to x=0.0, y=-3.0

dy -= 0.2;

if (dy <= -3.0)

{dy = -3.0;

frame = 6;

}

break;

case 6: //frame 1 polyman starts at right (7, -3) and walks to the middle right (3.5,-3)

//Polyman parameters

dx -= 0.15;

theta = theta - 5;

if (dx <= -7) {

dx = -7;

}

else {

frame = 7;

}

break;

case 7: //frame 1 polyman starts at right (7, -3) and walks to the middle right (3.5,-3)

//Polyman parameters

theta = theta + 5;

if (dx <= -7) {

dx = -7;

}

else {

frame = 6;

}

//we key on Polyman's position to change the frame

break;

}

// Redraw the scene with new coordinates

glutPostRedisplay();

glutTimerFunc(100, TimerFunction, 1);

}

