Nick Wright Problem 5 Solids CSC328

Code

#include<windows.h>

#include<GL/glut.h>

#include<stdlib.h>

#include<math.h>

#include<conio.h>

#include<stdio.h>

#include <iostream>

#include <iomanip>

using namespace std;

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HOMEWORK 5 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* NICK WRIGHT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The purpose of this program is to have a solid of polyman and polywoman walk onto the stage from opposite

sides. When polyman reaches the center of the screen he will open his mouth, jump and do a flip. Polywoman

will open her mouth and rock back and forth in approval. Both icons will then turn and walk off the left side of

the screen.

\*/

/\*-----------------Global Variables------------------\*/

//theta = global angular value for rotation

//dx and dy = global movement values for x and y, respectively

//POLYMAN GLOBAL VARIABLES --- He is starting on the right side of the screen

float bodyTheta = 0, bodyDX = 7.0, bodyDY = -3.0, bodyDZ = 1.0; //global values for the body

float mouthTheta = 0, mouthDX = 7.0, mouthDY = -3.0, mouthDZ = 1.0; //global values for the mouth

float leg1Theta = 0, leg1DX = 7.0, leg1DY = -3.0, leg1DZ = 1.0; //global value for leg 1

float leg2Theta = 0, leg2DX = 7.0, leg2DY = -3.0, leg2DZ = 1.0; //global value for leg 2

//POLYWOMAN GLOBAL VARIABLES --- She is starting on the left side of the screen

float rockTheta = 0, turnTheta = 180, PWbodyDX = -7.0, PWbodyDY = -3.0, PWbodyDZ = -1.0; //global values for the body

float PWmouthTheta = 0, mTurnTheta = 180, PWmouthDX = -7.0, PWmouthDY = -3.0, PWmouthDZ = -1.0; //global values for the mouth

float PWleg1Theta = 0, l1TurnTheta = 180, PWleg1DX = -7.0, PWleg1DY = -3.0, PWleg1DZ = -1.0; //global value for leg 1

float PWleg2Theta = 0, l2TurnTheta = 180, PWleg2DX = -7.0, PWleg2DY = -3.0, PWleg2DZ = -1.0; //global value for leg 2

float rockAccumulator = 0; //accumulates to break out of rock loop for polywoman (see case 5 and 6)

int frame = 1;

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

void RenderScene(void);//this is a function to draw the scene in an opened window

//\*\*\*CREATING THE LOADS AND DRAWS\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYMAN LOADS AND DRAWS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//body functions

void loadBody(float[][4], float[][4], float[][4], float[][4], float[][4], float[][4],

float[2], float[2], float[2]);

void drawBody(float[][4], float[][4], float[][4], float[][4], float[][4], float[][4],

float[2], float[2], float[2]);

//mouth functions

void loadMouth(float[][3], float[][3], float[][3]); //loads the mouth

void drawMouth(float[][3], float[][3], float[][3]); //draws the mouth

//leg functions

void drawLeg(float[], float[], float[]); //draws legs

void loadLeg1(float[], float[], float[]); //loads leg 1

void loadLeg2(float[], float[], float[]); //loads leg 2

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYWOMAN LOADS AND DRAWS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void PWloadBody(float[][4], float[][4], float[][4], float[][4], float[][4], float[][4],

float[2], float[2], float[2]);

void PWloadMouth(float[][3], float[][3], float[][3]); //loads the mouth

void PWloadLeg1(float[], float[], float[]); //loads leg 1

void PWloadLeg2(float[], float[], float[]); //loads leg 2

void PWloadDetails(float PWdetailX[][9], float PWdetailY[][9], float PWdetailZ[][9]); //loads the hair and bows

void PWdrawDetails(float PWdetailX[][9], float PWdetailY[][9], float PWdetailZ[][9]); //draws the hair and details

//\*\*\*CREATING THE MODELVIEW MATRICIES\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYMAN MODELVIEW\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void bodyModel(void); //sets the MODELVIEW MATRIX for the body and mouth (rotation/translation matrix)

void leg1Model(void); //sets the MODELVIEW MATRIX for leg 1 (rotation/translation matrix)

void leg2Model(void); //sets the MODELVIEW MATRIX for leg 2 (rotation/translation matrix)

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYWOMAN MODELVIEW\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void PWbodyModel(void); //sets the MODELVIEW MATRIX for the body and mouth (rotation/translation matrix)

void PWleg1Model(void); //sets the MODELVIEW MATRIX for leg 1 (rotation/translation matrix)

void PWleg2Model(void); //sets the MODELVIEW MATRIX for leg 2 (rotation/translation matrix)

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

void TimerFunction(int);

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

//Main Program

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

{

//set up the window title

char header[] = "Solids by Nick Wright (Homework 5)";

/\*glutInit() initializes GLUT. Takes the command line arguments which are used to

initialize the native window system.

This function must be called before any other GLUT functions.\*/

glutInit(&argc, argv);

//set up the display mode with a single buffer and rgb colors

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH);

SetupRC();

//initialize the window size and position

glutInitWindowSize(560, 440);

glutInitWindowPosition(140, 20);

//Initialize background color in window to red

// Open and Label Window

glutCreateWindow(header);

glutDisplayFunc(RenderScene);

glutTimerFunc(30, TimerFunction, 1);

//now draw the scene

glutMainLoop();

return 0;

}

//Render Scene Function

void RenderScene(void)

{

float xdel = 0.25;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYMAN PATTERN\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//pattern for polyman body

float upperX[2][4], upperY[2][4], upperZ[2][4], lowerX[2][4],

lowerY[2][4], lowerZ[2][4], eyeX[2], eyeY[2], eyeZ[2]; //polyman pattern

//pattern for mouth

float mX[2][3], mY[2][3], mZ[2][3];

//pattern for legs 1 and 2

float l1x[4], l1y[4], l1z[4];

float l2x[4], l2y[4], l2z[4];

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYWOMAN PATTERN\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//pattern for polywoman body

float PWupperX[2][4], PWupperY[2][4], PWupperZ[2][4], PWlowerX[2][4],

PWlowerY[2][4], PWlowerZ[2][4], PWeyeX[2], PWeyeY[2], PWeyeZ[2]; //polyman pattern;

//pattern for polywoman mouth

float PWmX[2][3], PWmY[2][3], PWmZ[2][3];

//pattern for polywoman legs 1 and 2

float PWl1x[4], PWl1y[4], PWl1z[4];

float PWl2x[4], PWl2y[4], PWl2z[4];

//pattern for polywoman details

float PWdetailsX[2][9], PWdetailsY[2][9], PWdetailsZ[2][9];

//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, 540, 440);

//Establish the clipping volume in user coordinates

glOrtho(-7.0, 7.0, -7.0, 7.0, 5.0, -5.0);

//load the icons untransformed

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYMAN LOADS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

loadBody(upperX, upperY, upperZ, lowerX, lowerY, lowerZ, eyeX, eyeY, eyeZ);

loadMouth(mX, mY, mZ);

loadLeg1(l1x, l1y, l1z);

loadLeg2(l2x, l2y, l2z);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYWOMAN LOADS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PWloadBody(PWupperX, PWupperY, PWupperZ, PWlowerX, PWlowerY, PWlowerZ, PWeyeX, PWeyeY, PWeyeZ);

PWloadMouth(PWmX, PWmY, PWmZ);

PWloadLeg1(PWl1x, PWl1y, PWl1z);

PWloadLeg2(PWl2x, PWl2y, PWl2z);

PWloadDetails(PWdetailsX, PWdetailsY, PWdetailsZ);

glEnable(GL\_DEPTH\_TEST);

glClearColor(0.5, 0.5, 0.5, 1.0);

//clear the window with the background color

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYMAN TRANFORMATIONS AND DRAWS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//glFlush being performed after each draw

bodyModel(); //body modelview matrix

drawBody(upperX, upperY, upperZ, lowerX, lowerY, lowerZ, eyeX, eyeY, eyeZ);

glFlush();

leg1Model(); //leg 1 modelview matrix

drawLeg(l1x, l1y, l1z);

glFlush();

leg2Model(); //leg 2 modelview matrix

drawLeg(l2x, l2y, l2z);

glFlush();

bodyModel();

//if the body is not at x = 0, the mouth will be drawn closed, else it will be drawn open

if (bodyDX != 0)

{

drawMouth(mX, mY, mZ);

glFlush();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYWOMAN TRANFORMATIONS AND DRAWS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//glFlush being performed after each draw

PWbodyModel(); //body modelview matrix

drawBody(PWupperX, PWupperY, PWupperZ, PWlowerX, PWlowerY, PWlowerZ, PWeyeX, PWeyeY, PWeyeZ);

glFlush();

PWleg1Model(); //leg 1 modelview matrix

drawLeg(PWl1x, PWl1y, PWl1z);

glFlush();

PWleg2Model(); //leg 2 modelview matrix

drawLeg(PWl2x, PWl2y, PWl2z);

glFlush();

PWbodyModel();

PWdrawDetails(PWdetailsX, PWdetailsY, PWdetailsZ);

glFlush();

PWbodyModel();

//if the body is not at x = 0, the mouth will be drawn closed, else it will be drawn open

if (bodyDX != 0)

{

drawMouth(PWmX, PWmY, PWmZ);

glFlush();

}

glFlush();

glEnd();

glutSwapBuffers();

return;

}//end of renderscene

//function timer

void TimerFunction(int value)

{

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

switch (frame)

{

case 1:

//frame 1 polyman starts at the right (7, -3) and walks to the middle (0,-3)

//body parameters

bodyDX -= 0.15;

//mouth parameters

mouthDX -= 0.15;

//leg parameters

leg1DX -= 0.15;

leg2DX -= 0.15;

//if else statement to make the legs move up and down

if (leg1DY > -3) {

leg1DY -= 0.1; leg2DY += 0.1;

}

else {

leg1DY += 0.1; leg2DY -= 0.1;

}

//polywoman walks in from the left

PWbodyDX += 0.075;

PWmouthDX += 0.075;

PWleg1DX += 0.075;

PWleg2DX += 0.075;

if (PWleg1DY > -3) {

PWleg1DY -= 0.1; PWleg2DY += 0.1;

}

else {

PWleg1DY += 0.1; PWleg2DY -= 0.1;

}

//use body position to change frame

if (bodyDX <= 0)

{

//polyman polywoman

bodyDX = 0; PWbodyDX = -3.5;

mouthDX = 0; PWmouthDX = -3.5;

leg1DX = 0; PWleg1DX = -3.5;

leg2DX = 0; PWleg2DX = -3.5;

leg1DY = -3.0; PWleg1DY = -3.0;

leg2DY = -3.0; PWleg2DY = -3.0;

frame = 2;

}

break;

case 2:

//frame 2 polyman opens his mouth and jumps into the air (y = 5)

//body parameters

bodyDY += 0.2;

//leg1 parameters

leg1DY += 0.2;

//leg2 parameters

leg2DY += 0.2;

//mouth parameters

mouthDY += 0.2;

if (bodyDY > 5.0)

{

bodyDY = 5.0;

leg1DY = 5.0;

leg2DY = 5.0;

mouthDY = 5.0;

frame = 3;

}

break;

case 3:

//frame 3 polyman rotates 360 degrees

//using negtaive theta value to make him do a backflip

//body parameters

bodyTheta -= 5.0;

//leg1 parameters

leg1Theta -= 5.0;

//leg2 parameters

leg2Theta -= 5.0;

//mouth parameters

mouthTheta -= 5.0;

if (bodyTheta <= -360.0)

{

frame = 4;

bodyTheta = 0.0;

leg1Theta = 0.0;

leg2Theta = 0.0;

mouthTheta = 0.0;

}

break;

case 4:

//frame 4 polyman lands back down on the ground (y = -3.0) polyman also closes his mouth

//body parameters

bodyDY -= 0.2;

//leg1 parameters

leg1DY -= 0.2;

//leg2 parameters

leg2DY -= 0.2;

//mouth parameters

mouthDY -= 0.2;

if (bodyDY <= -3.0)

{

bodyDY = -3.0;

mouthDY = -3.0;

leg1DY = -3.0;

leg2DY = -3.0;

frame = 5;

}

break;

case 5:

//polywoman rocks backwards

rockTheta += 5.0;

PWmouthTheta += 5.0;

PWleg1Theta += 5.0;

PWleg2Theta += 5.0;

rockAccumulator += 5.0; //using this as an accumulator so that polywoman only rocks a few times

if (rockAccumulator == 90)

{

rockTheta = 0.0;

PWmouthTheta = 0.0;

PWleg1Theta = 0.0;

PWleg2Theta = 0.0;

frame = 7;

break;

}

else if (rockTheta == 30)

{

frame = 6;

}

break;

case 6:

//polywoman rocks forwards

rockTheta -= 5.0;

PWmouthTheta -= 5.0;

PWleg1Theta -= 5.0;

PWleg2Theta -= 5.0;

if (rockTheta == -30)

{

frame = 5;

}

break;

case 7:

//polywoman rotates so that they may both walk off stage together

turnTheta -= 5.0;

mTurnTheta -= 5.0;

l1TurnTheta -= 5.0;

l2TurnTheta -= 5.0;

if (turnTheta <= 0)

{

turnTheta = 0;

mTurnTheta = 0;

l1TurnTheta = 0;

l2TurnTheta = 0;

frame = 8;

}

break;

case 8:

//frame 5 polyman walks off of the stage to the left

//polyman polywoman

bodyDX -= 0.15; PWbodyDX -= 0.15; //body parameters

mouthDX -= 0.15; PWmouthDX -= 0.15; //mouth parameters

leg1DX -= 0.15; PWleg1DX -= 0.15; //leg1 parameters

leg2DX -= 0.15; PWleg2DX -= 0.15; //leg2 parameters

//if else statement to make the legs move up and down

if (leg1DY > -3) {

leg1DY -= 0.1; leg2DY += 0.1;

}

else {

leg1DY += 0.1; leg2DY -= 0.1;

}

//if else statement to make polywomans legs move up and down

if (PWleg1DY > -3) {

PWleg1DY -= 0.1; PWleg2DY += 0.1;

}

else {

PWleg1DY += 0.1; PWleg2DY -= 0.1;

}

if (bodyDX <= -6.0)

{

//polyman polywoman

bodyDX = -6.5; PWbodyDX = -6.5;

mouthDX = -6.5; PWmouthDX = -6.5;

leg1DX = -6.5; PWleg1DX = -6.5;

leg2DX = -6.5; PWleg2DX = -6.5;

leg1DY = -3.0; PWleg1DY = -3.0;

leg2DY = -3.0; PWleg2DY = -3.0;

break;

}

}

// Redraw the scene with new coordinates

glutPostRedisplay();

glutTimerFunc(30, TimerFunction, 1);

}

//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

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYMAN FUNCTIONS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void loadMouth(float mX[][3], float mY[][3], float mZ[][3])

{

//load the front mouth

mX[0][0] = (-9.0 / 8); mY[0][0] = (0); mZ[0][0] = (1.0 / 2);

mX[0][1] = (-3.0 / 8); mY[0][1] = (0); mZ[0][1] = (1.0 / 2);

mX[0][2] = (-5.0 / 8); mY[0][2] = (-3.0 / 4); mZ[0][2] = (1.0 / 2);

//load the back mouth

mX[1][0] = (-9.0 / 8); mY[1][0] = (0); mZ[1][0] = (-1.0 / 2);

mX[1][1] = (-3.0 / 8); mY[1][1] = (0); mZ[1][1] = (-1.0 / 2);

mX[1][2] = (-5.0 / 8); mY[1][2] = (-3.0 / 4); mZ[1][2] = (-1.0 / 2);

}//end of loadMouth

void drawMouth(float mX[][3], float mY[][3], float mZ[][3])

{

//front mouth

glColor3f(1.0, 1.0, 0); //setting color to yellow

glFrontFace(GL\_CCW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glVertex3f(mX[0][0], mY[0][0], mZ[0][0]);

glVertex3f(mX[0][1], mY[0][1], mZ[0][1]);

glVertex3f(mX[0][2], mY[0][2], mZ[0][2]);

glEnd();

glFlush();

//back mouth

glColor3f(1.0, 1.0, 1.0); //setting color to white

glFrontFace(GL\_CW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glVertex3f(mX[1][0], mY[1][0], mZ[1][0]);

glVertex3f(mX[1][1], mY[1][1], mZ[1][1]);

glVertex3f(mX[1][2], mY[1][2], mZ[1][2]);

glEnd();

glFlush();

//side mouth

glColor3f(2.0, 0.5, 1.0);//lilac

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glVertex3f(mX[0][0], mY[0][0], mZ[0][0]);

glVertex3f(mX[1][0], mY[1][0], mZ[1][0]);

glVertex3f(mX[1][2], mY[1][2], mZ[1][2]);

glVertex3f(mX[0][2], mY[0][2], mZ[0][2]);

glEnd();

glFlush();

return;

}

void loadBody(float upperX[][4], float upperY[][4], float upperZ[][4], float lowerX[][4], float lowerY[][4],

float lowerZ[][4], float eyeX[2], float eyeY[2], float eyeZ[2]) {

//upper coords front face

upperX[0][0] = -9.0 / 8; upperY[0][0] = 0; upperZ[0][0] = 1.0 / 2;

upperX[0][1] = -5.0 / 8; upperY[0][1] = 3.0 / 4; upperZ[0][1] = 1.0 / 2;

upperX[0][2] = 5.0 / 8; upperY[0][2] = 3.0 / 4; upperZ[0][2] = 1.0 / 2;

upperX[0][3] = 9.0 / 8; upperY[0][3] = 0; upperZ[0][3] = 1.0 / 2;

//lower coords front face

lowerX[0][0] = -5.0 / 8; lowerY[0][0] = -3.0 / 4; lowerZ[0][0] = 1.0 / 2;

lowerX[0][1] = -3.0 / 8; lowerY[0][1] = 0; lowerZ[0][1] = 1.0 / 2;

lowerX[0][2] = 9.0 / 8; lowerY[0][2] = 0; lowerZ[0][2] = 1.0 / 2;

lowerX[0][3] = 5.0 / 8; lowerY[0][3] = -3.0 / 4; lowerZ[0][3] = 1.0 / 2;

//eye coordinates front face

eyeX[0] = -1.0 / 2; eyeY[0] = 1.0 / 2; eyeZ[0] = 1.0 / 2;

//upper coords back face

upperX[1][0] = -9.0 / 8; upperY[1][0] = 0; upperZ[1][0] = -1.0 / 2;

upperX[1][1] = -5.0 / 8; upperY[1][1] = 3.0 / 4; upperZ[1][1] = -1.0 / 2;

upperX[1][2] = 5.0 / 8; upperY[1][2] = 3.0 / 4; upperZ[1][2] = -1.0 / 2;

upperX[1][3] = 9.0 / 8; upperY[1][3] = 0; upperZ[1][3] = -1.0 / 2;

//lower coords back face

lowerX[1][0] = -5.0 / 8; lowerY[1][0] = -3.0 / 4; lowerZ[1][0] = -1.0 / 2;

lowerX[1][1] = -3.0 / 8; lowerY[1][1] = 0; lowerZ[1][1] = -1.0 / 2;

lowerX[1][2] = 9.0 / 8; lowerY[1][2] = 0; lowerZ[1][2] = -1.0 / 2;

lowerX[1][3] = 5.0 / 8; lowerY[1][3] = -3.0 / 4; lowerZ[1][3] = -1.0 / 2;

//eye coords back face

eyeX[1] = -1.0 / 2; eyeY[1] = 1.0 / 2; eyeZ[1] = -1.0 / 2;

return;

}//end of loadBody

void drawBody(float upperX[][4], float upperY[][4], float upperZ[][4], float lowerX[][4], float lowerY[][4],

float lowerZ[][4], float eyeX[2], float eyeY[2], float eyeZ[2])

{

int i;

glFrontFace(GL\_CW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

//front face

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(1.0, 1.0, 0.0);

for (i = 3; i >= 0; i--)

{

glVertex3f(upperX[0][i], upperY[0][i], upperZ[0][i]);

}

glEnd();

glFlush();

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(1.0, 1.0, 0.0);

for (i = 3; i >= 0; i--)

{

glVertex3f(lowerX[0][i], lowerY[0][i], lowerZ[0][i]);

}

glEnd();

glFlush();

glFrontFace(GL\_CCW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

//eye 1

glColor3f(0.0, 0.0, 0.0);

glPointSize(4);

glBegin(GL\_POINTS);

glVertex3f(eyeX[0], eyeY[0], eyeZ[0]);

glEnd();

glFlush();

//back face

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(1.0, 1.0, 1.0);

for (i = 3; i >= 0; i--)

{

glVertex3f(upperX[1][i], upperY[1][i], upperZ[1][i]);

}

glEnd();

glFlush();

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(1.0, 1.0, 1.0);

for (i = 3; i >= 0; i--)

{

glVertex3f(lowerX[1][i], lowerY[1][i], lowerZ[1][i]);

}

glEnd();

glFlush();

//back eye

glColor3f(0.0, 0.0, 0.0);

glPointSize(4);

glBegin(GL\_POINTS);

glVertex3f(eyeX[1], eyeY[1], eyeZ[1]);

glEnd();

glFlush();

//connecting pieces

//upper head

glFrontFace(GL\_CW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(0.5, 0.5, 1.0);

glVertex3f(upperX[0][1], upperY[0][1], upperZ[0][1]);

glVertex3f(upperX[1][1], upperY[1][1], upperZ[1][1]);

glVertex3f(upperX[1][0], upperY[1][0], upperZ[1][0]);

glVertex3f(upperX[0][0], upperY[0][0], upperZ[0][0]);

glEnd();

glFlush();

//top

glFrontFace(GL\_CW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(1, 0, 0); //red

glVertex3f(upperX[0][2], upperY[0][2], upperZ[0][2]);

glVertex3f(upperX[1][2], upperY[1][2], upperZ[1][2]);

glVertex3f(upperX[1][1], upperY[1][1], upperZ[1][1]);

glVertex3f(upperX[0][1], upperY[0][1], upperZ[0][1]);

glEnd();

glFlush();

//upper back

glFrontFace(GL\_CCW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(0.0, 1.0, 0.0); //bronze

glVertex3f(upperX[0][2], upperY[0][2], upperZ[0][2]);

glVertex3f(upperX[1][2], upperY[1][2], upperZ[1][2]);

glVertex3f(upperX[1][3], upperY[1][3], upperZ[1][3]);

glVertex3f(upperX[0][3], upperY[0][3], upperZ[0][3]);

glEnd();

glFlush();

//upper mouth

glFrontFace(GL\_CW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(0.0, 0.5, 0.5); //blue green

glVertex3f(upperX[0][0], upperY[0][0], upperZ[0][0]);

glVertex3f(upperX[1][0], upperY[1][0], upperZ[1][0]);

glVertex3f(upperX[1][3], upperY[1][3], upperZ[1][3]);

glVertex3f(upperX[0][3], upperY[0][3], upperZ[0][3]);

glEnd();

glFlush();

//lower mouth

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(1.0, 0.5, 0.0); //orange

glVertex3f(lowerX[0][1], lowerY[0][1], lowerZ[0][1]);

glVertex3f(lowerX[1][1], lowerY[1][1], lowerZ[1][1]);

glVertex3f(lowerX[1][0], lowerY[1][0], lowerZ[1][0]);

glVertex3f(lowerX[0][0], lowerY[0][0], lowerZ[0][0]);

glEnd();

glFlush();

//lower back

glFrontFace(GL\_CCW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(0.1, 0.0, 0.0); //brown

glVertex3f(lowerX[0][2], lowerY[0][2], lowerZ[0][2]);

glVertex3f(lowerX[1][2], lowerY[1][2], lowerZ[1][2]);

glVertex3f(lowerX[1][3], lowerY[1][3], lowerZ[1][3]);

glVertex3f(lowerX[0][3], lowerY[0][3], lowerZ[0][3]);

glEnd();

glFlush();

//bottom

glFrontFace(GL\_CW);

glCullFace(GL\_BACK);

glEnable(GL\_CULL\_FACE);

glShadeModel(GL\_SMOOTH);

glBegin(GL\_POLYGON);

glColor3f(0.5, 1.0, 1.0); //cyan

glVertex3f(lowerX[0][0], lowerY[0][0], lowerZ[0][0]);

glVertex3f(lowerX[1][0], lowerY[1][0], lowerZ[1][0]);

glVertex3f(lowerX[1][3], lowerY[1][3], lowerZ[1][3]);

glVertex3f(lowerX[0][3], lowerY[0][3], lowerZ[0][3]);

glEnd();

glFlush();

return;

}//end of drawbody

void loadLeg1(float l1x[], float l1y[], float l1z[])

{

//this function will load leg 1

l1x[0] = -1.0 / 4; l1y[0] = -1.0 / 2; l1z[0] = 1.0 / 2;

l1x[1] = -1.0 / 4; l1y[1] = -1.0; l1z[1] = 1.0 / 2;

l1x[2] = -1.0 / 2; l1y[2] = -1.0; l1z[2] = 1.0 / 2;

//cyan color

l1x[3] = 0; l1y[3] = 1.0; l1z[3] = 1.0;

return;

}//end of loadLeg1

void drawLeg(float l1x[], float l1y[], float l1z[])

{

//this function will draw leg 1

//setting color

glColor3f(l1x[3], l1y[3], l1z[3]);

glBegin(GL\_LINE\_STRIP);

glVertex3f(l1x[0], l1y[0], l1z[0]);

glVertex3f(l1x[1], l1y[1], l1z[1]);

glVertex3f(l1x[2], l1y[2], l1z[2]);

glEnd();

glFlush();

return;

}//end of drawLeg1

void loadLeg2(float l2x[], float l2y[], float l2z[])

{

//this function will load leg 2

l2x[0] = 1.0 / 4; l2y[0] = -1.0 / 2; l2z[0] = -1.0 / 2;

l2x[1] = 1.0 / 4; l2y[1] = -1.0; l2z[1] = -1.0 / 2;

l2x[2] = 0; l2y[2] = -1.0; l2z[2] = -1.0 / 2;

//blue color

l2x[3] = 0; l2y[3] = 0; l2z[3] = 1.0;

return;

}//end of loadLeg1

//function bodyModel

void bodyModel()

{

//float bodyTheta = 0, bodyDX = -6.0, bodyDY = -3.0;

//sets the modelviel matrix for the body

cout << "in bodyModel" << endl;

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(bodyDX, bodyDY, bodyDZ);

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

glRotatef(30.0, 1.0, 0.0, 0.0);

return;

}//end of bodyModel

//function leg1Model

void leg1Model()

{

//float leg1Theta = 0, leg1DX = -6.0, leg1DY = -3.0;

//sets the modelviel matrix for leg1

cout << "in leg1Model" << endl;

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(leg1DX, leg1DY, leg1DZ);

glRotatef(leg1Theta, 0.0, 0.0, 1.0);// note that the angle theta is in degrees, not radians

glRotatef(30.0, 1.0, 0.0, 0.0);

return;

}//end of leg1Model

//function leg2Model

void leg2Model()

{

//float leg2Theta = 0, leg2DX = -6.0, leg2DY = -3.0;

//sets the modelviel matrix for leg2

cout << "in leg2Model" << endl;

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(leg2DX, leg2DY, leg2DZ);

glRotatef(leg2Theta, 0.0, 0.0, 1.0);// note that the angle theta is in degrees, not radians

glRotatef(30.0, 1.0, 0.0, 0.0);

return;

}//end of leg2Model

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*POLYWOMAN FUNCTIONS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void PWloadBody(float upperX[][4], float upperY[][4], float upperZ[][4], float lowerX[][4], float lowerY[][4],

float lowerZ[][4], float eyeX[2], float eyeY[2], float eyeZ[2]) {

//upper coords front face

upperX[0][0] = -9.0 / 8; upperY[0][0] = 0; upperZ[0][0] = 1.0 / 2;

upperX[0][1] = -5.0 / 8; upperY[0][1] = 3.0 / 4; upperZ[0][1] = 1.0 / 2;

upperX[0][2] = 5.0 / 8; upperY[0][2] = 3.0 / 4; upperZ[0][2] = 1.0 / 2;

upperX[0][3] = 9.0 / 8; upperY[0][3] = 0; upperZ[0][3] = 1.0 / 2;

//lower coords front face

lowerX[0][0] = -5.0 / 8; lowerY[0][0] = -3.0 / 4; lowerZ[0][0] = 1.0 / 2;

lowerX[0][1] = -3.0 / 8; lowerY[0][1] = 0; lowerZ[0][1] = 1.0 / 2;

lowerX[0][2] = 9.0 / 8; lowerY[0][2] = 0; lowerZ[0][2] = 1.0 / 2;

lowerX[0][3] = 5.0 / 8; lowerY[0][3] = -3.0 / 4; lowerZ[0][3] = 1.0 / 2;

//eye coordinates front face

eyeX[0] = -1.0 / 2; eyeY[0] = 1.0 / 2; eyeZ[0] = 1.0 / 2;

//upper coords back face

upperX[1][0] = -9.0 / 8; upperY[1][0] = 0; upperZ[1][0] = -1.0 / 2;

upperX[1][1] = -5.0 / 8; upperY[1][1] = 3.0 / 4; upperZ[1][1] = -1.0 / 2;

upperX[1][2] = 5.0 / 8; upperY[1][2] = 3.0 / 4; upperZ[1][2] = -1.0 / 2;

upperX[1][3] = 9.0 / 8; upperY[1][3] = 0; upperZ[1][3] = -1.0 / 2;

//lower coords back face

lowerX[1][0] = -5.0 / 8; lowerY[1][0] = -3.0 / 4; lowerZ[1][0] = -1.0 / 2;

lowerX[1][1] = -3.0 / 8; lowerY[1][1] = 0; lowerZ[1][1] = -1.0 / 2;

lowerX[1][2] = 9.0 / 8; lowerY[1][2] = 0; lowerZ[1][2] = -1.0 / 2;

lowerX[1][3] = 5.0 / 8; lowerY[1][3] = -3.0 / 4; lowerZ[1][3] = -1.0 / 2;

//eye coords back face

eyeX[1] = -1.0 / 2; eyeY[1] = 1.0 / 2; eyeZ[1] = -1.0 / 2;

return;

} //end of polywoman load body

void PWloadMouth(float PWmX[][3], float PWmY[][3], float PWmZ[][3])

{

//load the front mouth

PWmX[0][0] = (-9.0 / 8); PWmY[0][0] = (0); PWmZ[0][0] = (1.0 / 2);

PWmX[0][1] = (-3.0 / 8); PWmY[0][1] = (0); PWmZ[0][1] = (1.0 / 2);

PWmX[0][2] = (-5.0 / 8); PWmY[0][2] = (-3.0 / 4); PWmZ[0][2] = (1.0 / 2);

//load the back mouth

PWmX[1][0] = (-9.0 / 8); PWmY[1][0] = (0); PWmZ[1][0] = (-1.0 / 2);

PWmX[1][1] = (-3.0 / 8); PWmY[1][1] = (0); PWmZ[1][1] = (-1.0 / 2);

PWmX[1][2] = (-5.0 / 8); PWmY[1][2] = (-3.0 / 4); PWmZ[1][2] = (-1.0 / 2);

}//end of loadMouth

void PWloadLeg1(float PWl1x[], float PWl1y[], float PWl1z[])

{

//this function will load leg 1

PWl1x[0] = -1.0 / 4; PWl1y[0] = -1.0 / 2; PWl1z[0] = 1.0 / 2;

PWl1x[1] = -1.0 / 4; PWl1y[1] = -1.0; PWl1z[1] = 1.0 / 2;

PWl1x[2] = -1.0 / 2; PWl1y[2] = -1.0; PWl1z[2] = 1.0 / 2;

//leg1 color (lilac)

PWl1x[3] = 2.0; PWl1y[3] = 0.5; PWl1z[3] = 1.0;

return;

}//end of loadLeg1

void PWloadLeg2(float PWl2x[], float PWl2y[], float PWl2z[])

{

//this function will load leg 2

PWl2x[0] = 1.0 / 4; PWl2y[0] = -1.0 / 2; PWl2z[0] = -1.0 / 2;

PWl2x[1] = 1.0 / 4; PWl2y[1] = -1.0; PWl2z[1] = -1.0 / 2;

PWl2x[2] = 0; PWl2y[2] = -1.0; PWl2z[2] = -1.0 / 2;

//leg2 color (purple)

PWl2x[3] = 0.7; PWl2y[3] = 0; PWl2z[3] = 0.7;

return;

}//end of loadLeg1

//function PWbodyModel

void PWbodyModel()

{

//sets the modelviel matrix for the body

cout << "in Poly Woman bodyModel" << endl;

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(PWbodyDX, PWbodyDY, PWbodyDZ);

// note that the angle theta is in degrees, not radians

glRotatef(rockTheta, 0.0, 0.0, 1.0); //rotation for rocking movement

glRotatef(turnTheta, 0.0, 1.0, 0.0); //rotation for turning

glRotatef(30.0, 1.0, 0.0, 0.0);

return;

}//end of bodyModel

//function leg1Model

void PWleg1Model()

{

//sets the modelviel matrix for leg1

cout << "in leg1Model" << endl;

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(PWleg1DX, PWleg1DY, PWleg1DZ);

// note that the angle theta is in degrees, not radians

glRotatef(PWleg1Theta, 0.0, 0.0, 1.0);

glRotatef(l1TurnTheta, 0.0, 1.0, 0.0);

glRotatef(30.0, 1.0, 0.0, 0.0);

return;

}//end of leg1Model

//function leg2Model

void PWleg2Model()

{

//sets the modelviel matrix for leg2

cout << "in leg2Model" << endl;

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(PWleg2DX, PWleg2DY, PWleg2DZ);

// note that the angle theta is in degrees, not radians

glRotatef(PWleg2Theta, 0.0, 0.0, 1.0);

glRotatef(l2TurnTheta, 0.0, 1.0, 0.0);

glRotatef(30.0, 1.0, 0.0, 0.0);

return;

}//end of leg2Model

void PWloadDetails(float PWdetailX[][9], float PWdetailY[][9], float PWdetailZ[][9])

{

//polywoman front details

//color (lilac)

PWdetailX[0][0] = 2.0; PWdetailY[0][0] = 0.5; PWdetailZ[0][0] = 1.0;

//hair

PWdetailX[0][1] = -1.0 / 4; PWdetailY[0][1] = 3.0 / 4; PWdetailZ[0][1] = 1.0 / 2;

PWdetailX[0][2] = -1.0 / 4; PWdetailY[0][2] = 1.0 / 4; PWdetailZ[0][2] = 1.0 / 2;

PWdetailX[0][3] = 0.0; PWdetailY[0][3] = 3.0 / 4; PWdetailZ[0][3] = 1.0 / 2;

PWdetailX[0][4] = 0.0; PWdetailY[0][4] = 1.0 / 4; PWdetailZ[0][4] = 1.0 / 2;

PWdetailX[0][5] = 1.0 / 4; PWdetailY[0][5] = 3.0 / 4; PWdetailZ[0][5] = 1.0 / 2;

PWdetailX[0][6] = 1.0 / 4; PWdetailY[0][6] = 1.0 / 4; PWdetailZ[0][6] = 1.0 / 2;

PWdetailX[0][7] = 1.0 / 2; PWdetailY[0][7] = 3.0 / 4; PWdetailZ[0][7] = 1.0 / 2;

PWdetailX[0][8] = 1.0 / 2; PWdetailY[0][8] = 1.0 / 4; PWdetailZ[0][8] = 1.0 / 2;

//polywoman rear details

//color (purple)

PWdetailX[1][0] = 0.7; PWdetailY[1][0] = 0.0; PWdetailZ[1][0] = 0.7;

//hair

PWdetailX[1][1] = -1.0 / 4; PWdetailY[1][1] = 3.0 / 4; PWdetailZ[1][1] = -1.0 / 2;

PWdetailX[1][2] = -1.0 / 4; PWdetailY[1][2] = 1.0 / 4; PWdetailZ[1][2] = -1.0 / 2;

PWdetailX[1][3] = 0.0; PWdetailY[1][3] = 3.0 / 4; PWdetailZ[1][3] = -1.0 / 2;

PWdetailX[1][4] = 0.0; PWdetailY[1][4] = 1.0 / 4; PWdetailZ[1][4] = -1.0 / 2;

PWdetailX[1][5] = 1.0 / 4; PWdetailY[1][5] = 3.0 / 4; PWdetailZ[1][5] = -1.0 / 2;

PWdetailX[1][6] = 1.0 / 4; PWdetailY[1][6] = 1.0 / 4; PWdetailZ[1][6] = -1.0 / 2;

PWdetailX[1][7] = 1.0 / 2; PWdetailY[1][7] = 3.0 / 4; PWdetailZ[1][7] = -1.0 / 2;

PWdetailX[1][8] = 1.0 / 2; PWdetailY[1][8] = 1.0 / 4; PWdetailZ[1][8] = -1.0 / 2;

}

void PWdrawDetails(float PWdetailX[][9], float PWdetailY[][9], float PWdetailZ[][9])

{

//draw connecting pieces across top of head

glColor3f(1.0, 1.0, 1.0); //color white

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][1], PWdetailY[0][1], PWdetailZ[0][1]);

glVertex3f(PWdetailX[1][1], PWdetailY[1][1], PWdetailZ[1][1]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][3], PWdetailY[0][3], PWdetailZ[0][3]);

glVertex3f(PWdetailX[1][3], PWdetailY[1][3], PWdetailZ[1][3]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][5], PWdetailY[0][1], PWdetailZ[0][1]);

glVertex3f(PWdetailX[1][5], PWdetailY[1][1], PWdetailZ[1][1]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][7], PWdetailY[0][7], PWdetailZ[0][7]);

glVertex3f(PWdetailX[1][7], PWdetailY[1][7], PWdetailZ[1][7]);

glEnd(); glFlush();

//draw back

glColor3f(PWdetailX[1][0], PWdetailY[1][0], PWdetailZ[1][0]);

glBegin(GL\_LINES);

glVertex3f(PWdetailX[1][1], PWdetailY[1][1], PWdetailZ[1][1]);

glVertex3f(PWdetailX[1][2], PWdetailY[1][2], PWdetailZ[1][2]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[1][3], PWdetailY[1][3], PWdetailZ[1][3]);

glVertex3f(PWdetailX[1][4], PWdetailY[1][4], PWdetailZ[1][4]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[1][5], PWdetailY[1][5], PWdetailZ[1][5]);

glVertex3f(PWdetailX[1][6], PWdetailY[1][6], PWdetailZ[1][6]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[1][7], PWdetailY[1][7], PWdetailZ[1][7]);

glVertex3f(PWdetailX[1][8], PWdetailY[1][8], PWdetailZ[1][8]);

glEnd(); glFlush();

//draw front

glColor3f(PWdetailX[0][0], PWdetailY[0][0], PWdetailZ[0][0]);

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][1], PWdetailY[0][1], PWdetailZ[0][1]);

glVertex3f(PWdetailX[0][2], PWdetailY[0][2], PWdetailZ[0][2]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][3], PWdetailY[0][3], PWdetailZ[0][3]);

glVertex3f(PWdetailX[0][4], PWdetailY[0][4], PWdetailZ[0][4]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][5], PWdetailY[0][5], PWdetailZ[0][5]);

glVertex3f(PWdetailX[0][6], PWdetailY[0][6], PWdetailZ[0][6]);

glEnd(); glFlush();

glBegin(GL\_LINES);

glVertex3f(PWdetailX[0][7], PWdetailY[0][7], PWdetailZ[0][7]);

glVertex3f(PWdetailX[0][8], PWdetailY[0][8], PWdetailZ[0][8]);

glEnd(); glFlush();

//drawing some bows

//PWdetailX[0][0] = 2.0; PWdetailY[0][0] = 0.5; PWdetailZ[0][0] = 1.0;

glColor3f(2.0, 0.5, 1.0);

glPointSize(4);

glBegin(GL\_POINTS);

glVertex3f(PWdetailX[0][2], PWdetailY[0][2], PWdetailZ[0][2]);

glVertex3f(PWdetailX[1][2], PWdetailY[1][2], PWdetailZ[1][2]);

glVertex3f(PWdetailX[0][4], PWdetailY[0][4], PWdetailZ[0][4]);

glVertex3f(PWdetailX[1][4], PWdetailY[1][4], PWdetailZ[1][4]);

glVertex3f(PWdetailX[0][6], PWdetailY[0][6], PWdetailZ[0][6]);

glVertex3f(PWdetailX[1][6], PWdetailY[1][6], PWdetailZ[1][6]);

glVertex3f(PWdetailX[0][8], PWdetailY[0][8], PWdetailZ[0][8]);

glVertex3f(PWdetailX[1][8], PWdetailY[1][8], PWdetailZ[1][8]);

glEnd();

glFlush();

}

Output

