Daniel Meyer

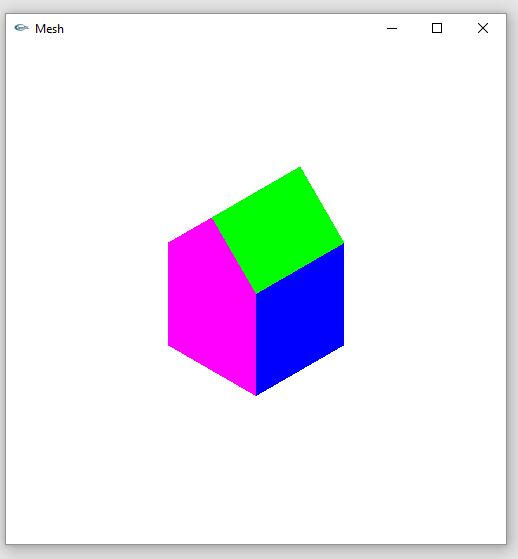
CSE 420-01

Lab 7

Polygon Modeling

**Lab 7 Report**

**Part 1: (success)**



void Mesh::setColor( int n )

{

if ( n == 1 )

glColor3f( 1, 0, 0 );

else if ( n == 2 )

glColor3f( 0, 1, 0 );

else if ( n == 3 )

glColor3f( 0, 0, 1 );

else if ( n == 4 )

glColor3f( 1, 1, 0 );

else if ( n == 5 )

glColor3f( 1, 0, 1 );

else if ( n == 6 )

glColor3f( 0, 1, 1 );

else

glColor3f( 0, 0, 0 );

}

void Mesh::drawMesh() // use OpenGL to draw this mesh

{

// draw each face of this mesh using OpenGL: draw each polygon.

if( isEmpty() ) return; // mesh is empty

glEnable( GL\_CULL\_FACE );

glCullFace ( GL\_BACK );

for(int f = 0; f < numFaces; f++) // draw each face

//for(int f = 6; f < numFaces; f++) // draw each face

{

glBegin(GL\_POLYGON);

cout << endl;

setColor( f );

for(int v = 0; v < face[f].nVerts; v++) // for each vertex

{

int in = face[f].vert[v].normIndex ; // index of this normal

int iv = face[f].vert[v].vertIndex ; // index of this vertex

glNormal3f(norm[in].x, norm[in].y, norm[in].z);

cout << "[" << norm[in].x << "," << norm[in].y << "," <<

norm[in].z << "]" << " ";

glVertex3f(pt[iv].x, pt[iv].y, pt[iv].z);

cout << "(" << pt[iv].x << "," << pt[iv].y << "," <<

pt[iv].z << ")" << " ";

}

glEnd();

//SDL\_Delay ( 1000 );

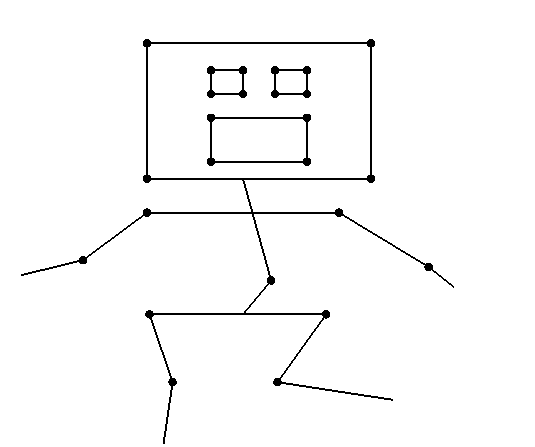
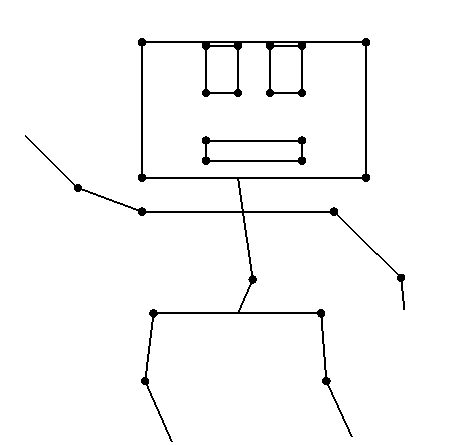
glFlush ();

cout << endl;

}

} //drawMesh

**Part 2: (success)**



void makeHead(Point2 A[], Point2 B[])

{

A[0].x = 3; A[0].y = 8; A[1].x = 6.5; A[1].y = 8;

A[2].x = 6.5; A[2].y = 12; A[3].x = 3; A[3].y = 12;

A[4].x = 3; A[4].y = 8; A[5].x = 6.5; A[5].y = 8;

B[0].x = 3; B[0].y = 8; B[1].x = 6.5; B[1].y = 8;

B[2].x = 6.5; B[2].y = 12; B[3].x = 3; B[3].y = 12; //Head

B[4].x = 3; B[4].y = 8; B[5].x = 6.5; B[5].y = 8;

}

void makeMouth(Point2 A[], Point2 B[])

{

A[0].x = 4; A[0].y = 8.5; A[1].x = 5.5; A[1].y = 8.5;

A[2].x = 5.5; A[2].y = 9; A[3].x = 4; A[3].y = 9;

A[4].x = 4; A[4].y = 8.5; A[5].x = 5.5; A[5].y = 8.5;

B[0].x = 4; B[0].y = 8.5; B[1].x = 5.5; B[1].y = 8.5;

B[2].x = 5.5; B[2].y = 10; B[3].x = 4; B[3].y = 10; //Mouth

B[4].x = 4; B[4].y = 8.5; B[5].x = 5.5; B[5].y = 8.5;

}

void makeLeftEye(Point2 A[], Point2 B[])

{

A[0].x = 4; A[0].y = 10.5; A[1].x = 4.5; A[1].y = 10.5;

A[2].x = 4.5; A[2].y = 12; A[3].x = 4; A[3].y = 12;

A[4].x = 4; A[4].y = 10.5; A[5].x = 4.5; A[5].y = 10.5;

B[0].x = 4; B[0].y = 10.5; B[1].x = 4.5; B[1].y = 10.5;

B[2].x = 4.5; B[2].y = 11; B[3].x = 4; B[3].y = 11;

B[4].x = 4; B[4].y = 10.5; B[5].x = 4.5; B[5].y = 10.5;

}

void makeRightEye(Point2 A[], Point2 B[])

{

A[0].x = 5; A[0].y = 10.5; A[1].x = 5.5; A[1].y = 10.5;

A[2].x = 5.5; A[2].y = 12; A[3].x = 5; A[3].y = 12;

A[4].x = 5; A[4].y = 10.5; A[5].x = 5.5; A[5].y = 10.5;

B[0].x = 5; B[0].y = 10.5; B[1].x = 5.5; B[1].y = 10.5;

B[2].x = 5.5; B[2].y = 11; B[3].x = 5; B[3].y = 11;

B[4].x = 5; B[4].y = 10.5; B[5].x = 5.5; B[5].y = 10.5;

}

Point2 A[10], B[10];

Point2 A1[10], B1[10];

Point2 A2[10], B2[10];

Point2 A3[10], B3[10];

Point2 A4[10], B4[10];

Point2 A5[10], B5[10];

Point2 A6[10], B6[10];

Point2 center(0, 0);

float t = 0, deltat = 0.1;

float deltax = 2, deltay = 0;

void init(void)

{

gluOrtho2D(0.0, 30.0, 0.0, 30.0);

makeHand(A, B); //create figure A and B

makeBody(A1, B1); //create figure A1 and B1

makeLeg(A2, B2); //create figure A2 and B2

makeHead(A3, B3);

makeMouth(A4, B4);

makeLeftEye(A5, B5);

makeRightEye(A6, B6);

glLineWidth(2);

glClearColor(1.0, 1.0, 1.0, 0.0);

}

void display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

drawTween(A, B, 6, t, center);

drawTween(A1, B1, 3, t, center);

drawTween(A2, B2, 6, t, center);

drawTween(A3, B3, 6, t, center);

drawTween(A4, B4, 6, t, center);

drawTween(A5, B5, 6, t, center);

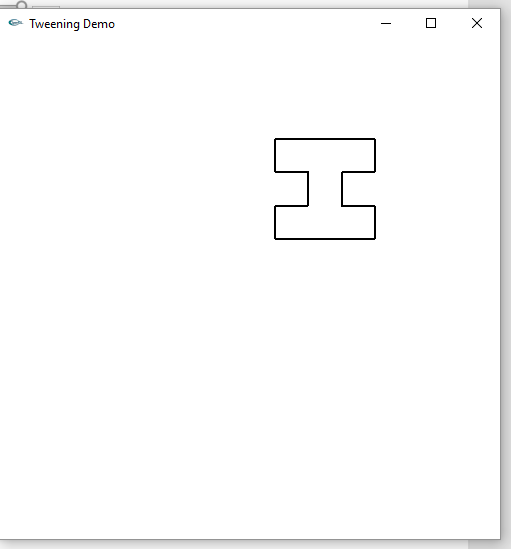
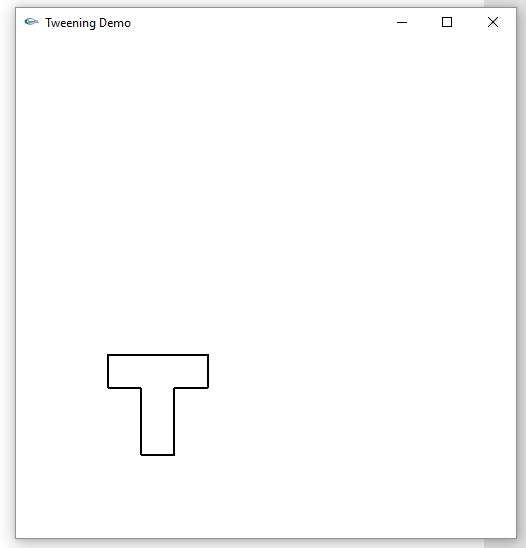
drawTween(A6, B6, 6, t, center);

glFlush();

glutSwapBuffers();

}

**Extra Credit: (Success)**



void makeFigures(Point2 A[], Point2 B[])

{

A[0].x = 0; A[0].y = 0; A[1].x = 3; A[1].y = 0;

A[2].x = 6; A[2].y = 0; A[3].x = 6; A[3].y = -2;

A[4].x = 4; A[4].y = -2; A[5].x = 4; A[5].y = -4;

A[6].x = 6; A[6].y = -4; A[7].x = 6; A[7].y = -6;

A[8].x = 0; A[8].y = -6; A[9].x = 0; A[9].y = -4;

A[10].x = 2; A[10].y = -4; A[11].x = 2; A[11].y = -2;

A[12].x = 0; A[12].y = -2;

B[0].x = 0; B[0].y = 0; B[1].x = 3; B[1].y = 0;

B[2].x = 6; B[2].y = 0; B[3].x = 6; B[3].y = -2;

B[4].x = 4; B[4].y = -2; B[5].x = 4; B[5].y = -6;

B[6].x = 2; B[6].y = -6; B[7].x = 2; B[7].y = -2;

B[8].x = 0; B[8].y = -2;

}

void setColor(int i)

{

switch (i) {

case 1:

glColor3f(0, 1, 1);

break;

case 2:

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

break;

case 3:

glColor3f(1, 1, 0);

break;

case 4:

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

break;

case 5:

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

break;

default:

glColor3f(0, 0, 0); //black

}

}

**Summary:**

For the first part of the assignment I had to modify the given mesh.cpp file to use a different set of data. I did this by modifying the data.txt file’s contents and then changed the colors of the new barn object. Then, per the instructions, I added the code for culling to increase the clarity of the barn by culling the back faces. For the next part of the assignment, I added the head, mouth, and eyes to the animated skeleton. I then animated the eyes and mouth using tweening. Finally, for the extra credit I modified another of the provided programs to animate the morphing of a T into an I in 10 steps. All of the programs compiled and ran correctly. As a result, I feel I earned the full 20 points and the 5 points extra credit.