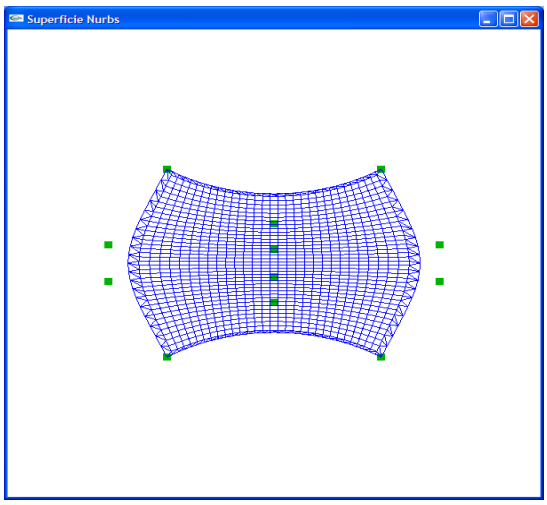
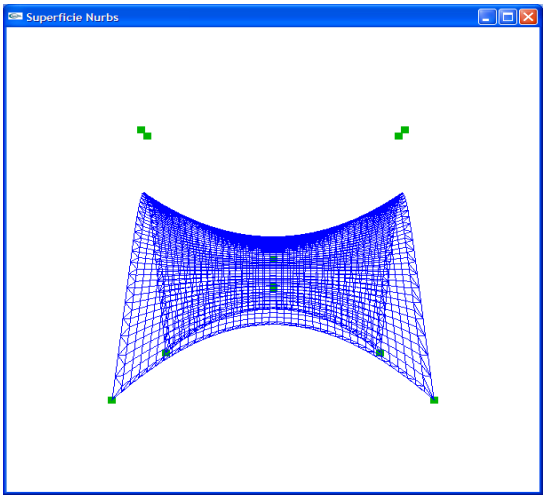
Código 20: Superficie NURB

```
#include <windows.h>
#include <GL/glut.h>

GLUnurbsObj *Nurb1;
GLboolean filledFlag=GL_FALSE;
GLfloat girax = 0, giray= 90, zoom= -40, updw= 0, izde= 0;

GLfloat ctlpts1[4][3][3]= {
                          {{-8, -8, 8}, {-2.8,0, 0}, {-8, -8, -8} },  //u=0  v=0,1,2
                          {{ 1, 9, 8} , {-1, -2, 0},  { 1, 9, -8} },  //u=1  v=0,1,2
                          {{-1, 9, 8} , { 1, -2, 0},  {-1, 9, -8} },  //u=2  v=0,1,2
                          {{ 8, -8, 8}, { 2.8,0, 0}, { 8, -8, -8} }   //u=3  v=0,1,2
};
// >>>>>>>>>>>>>>>>>>>>>> Rotacion y Zoom >>>>>>>>>>>>>>>>>>>>>>
void mover (void)
{
     glTranslated( 0, 0, zoom);
     glRotatef( girax, 1.0, 0.0, 0.0);
     glRotatef( giray, 0.0, 1.0, 0.0);
}
// >>>>>>>>>>>>>>>>>>  Funciones con Teclado >>>>>>>>>>>>>>>>>
void teclado(unsigned char key, int x, int y) {
     switch(key) {
          case 27:  exit (0);
                    if (Nurb1) gluDeleteNurbsRenderer(Nurb1);
                    break;
          case 'i': zoom += 5; //zoom in
                    break;
          case 'o': zoom -= 5; //zoom out
                    break;
          case 'f': filledFlag = !filledFlag;
          default : break;
     }
     glutPostRedisplay();
}
void flechas(int key, int x, int y) {
     switch(key) {
          case GLUT_KEY_LEFT:   giray-= 15;         //giro izq y
                     break;
          case GLUT_KEY_RIGHT:  giray+= 15;         //giro der y
                     break;
          case GLUT_KEY_UP:     girax -= 15;        //giro arriba x
                     break;
          case GLUT_KEY_DOWN:   girax += 15;        //giro abajo x
          default:break;
     }
     glutPostRedisplay();
}
// <<<<<<<<<<<<<<<<<<<<<< inicializa  >>>>>>>>>>>>>>>>>>
void inicializa(void)
{
     Nurb1 = gluNewNurbsRenderer();
     gluNurbsProperty(Nurb1, GLU_SAMPLING_TOLERANCE, 25.0);
}
//<<<<<<<<<<<<<<<<<<<<<<<< Dibuja >>>>>>>>>>>>>>>>>>
void dibuja(void)
{
     int u,v;
     GLfloat knots[8] = {0.0, 1.0, 1.0, 1.0, 2.0, 2.0, 2.0, 3.0};
     GLfloat knots2[6] = {0.0, 1.0, 1.0, 2.0, 2.0, 3.0};

     glClear(GL_COLOR_BUFFER_BIT);

     glPushMatrix();
        mover();
        glPointSize(9.0);
        glColor3f(0.0, 0.7, 0.0);
```

```
        glBegin (GL_POINTS);              //dibuja puntos de control
           for (u=0; u<4; u++)
              for (v=0; v<3;v++)
                  glVertex3f(ctlpts1[u][v][0], ctlpts1[u][v][1], ctlpts1[u][v][2]);
        glEnd();
        //dibuja nurb
        glColor3f(0.0, 0.0, 1.0);
        if (filledFlag == GL_FALSE)
            gluNurbsProperty(Nurb1, GLU_DISPLAY_MODE, GLU_OUTLINE_POLYGON);
         else
            gluNurbsProperty(Nurb1, GLU_DISPLAY_MODE, GLU_FILL);
         gluBeginSurface(Nurb1);
             gluNurbsSurface(Nurb1,8, knots, 6, knots2, 3*3, 3,
                             &ctlpts1[0][0][0], 4, 3, GL_MAP2_VERTEX_3);
         gluEndSurface(Nurb1);
     glPopMatrix();
     glutSwapBuffers();
}
// <<<<<<<<<<<<<<<<<<<<<<< Reshape >>>>>>>>>>>>>>>>>>>>>>>
void reshape(int w, int h)
{
     glClearColor(1.0,1.0,1.0,0.0);
     glViewport(0, 0, w, h);
     glMatrixMode(GL_PROJECTION);
     glLoadIdentity();
     gluPerspective(45, (float)w/h, 10, 150);
     glMatrixMode(GL_MODELVIEW);
     glLoadIdentity();
}
// <<<<<<<<<<<<<<<<<<<<<<< main >>>>>>>>>>>>>>>>>>>>>>>
int main(int argc, char** argv)
{
     glutInit(&argc, argv);
     glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);
     glutInitWindowPosition(0, 0);
     glutInitWindowSize(600,600);
     glutCreateWindow("Superficie Nurbs");
     inicializa();
     glutDisplayFunc(dibuja);
     glutReshapeFunc(reshape);
     glutKeyboardFunc(teclado);
     glutSpecialFunc(flechas);
     glutMainLoop();
     return 0;
}
```