

Código 17: Textura sintetica (frag)

```
#include <GL/glut.h>
#include <GL/glaux.h>
#include <stdio.h>

/* GIMP RGB C-Source image dump (woodfloor2.c) */
static const struct {
    unsigned int width;
    unsigned int height;
    unsigned int bytes_per_pixel; /* 3:RGB, 4:RGBA */
    char pixel_data[128 * 128 * 3 + 1];
} gimp_image = {
    128, 128, 3,
    "\274\1779\264v4\264u8\260p7\25011\24710\257x9\240k,\251t/\255s3\224^1\
217"
    ... };
GLuint texture[2]; //2 texturas

AUX_RGBImageRec *LoadBMP(char *Filename)
{
    FILE *File=NULL;
    if (!Filename) return NULL;
    File=fopen(Filename,"r");
    if (File) {
        fclose(File);
        return auxDIBImageLoad(Filename); // carga Bitmap y regresa apuntador
    }
    return NULL; // Si falla la carga regresa NULL
}

void LoadGLTextures()
{
    AUX_RGBImageRec *TextureImage[1];
    memset(TextureImage,0,sizeof(void *)*1); // Inicializa apuntador en NULL

    if (TextureImage[0]=LoadBMP("PuestaSol.bmp")) {
        glBindTexture(GL_TEXTURE_2D,texture[0]);
        glTexImage2D(GL_TEXTURE_2D, 0, 3, TextureImage[0]->sizeX,TextureImage[0]->sizeY,
            0, GL_RGB, GL_UNSIGNED_BYTE, TextureImage[0]->data);
        glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
        glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
    }
}

void CreaTextura()
{
    glEnable(GL_TEXTURE_2D);
    glBindTexture(GL_TEXTURE_2D, texture[1]);
    glTexImage2D(GL_TEXTURE_2D, 0, 3, gimp_image.width, gimp_image.height,
        0, GL_RGB, GL_UNSIGNED_BYTE, gimp_image.pixel_data);
    glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
}

void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glGenTextures(2, texture);
    CreaTextura();
    LoadGLTextures();
    glBindTexture(GL_TEXTURE_2D,texture[0]);
    glBegin(GL_QUADS);
        glTexCoord2f(0.0,0.0); glVertex3f(-2.0,-2.0,0.0);
        glTexCoord2f(1.0,0.0); glVertex3f(2.0,-2.0,0.0);
        glTexCoord2f(1.0,1.0); glVertex3f(2.0,2.0,0.0);
        glTexCoord2f(0.0,1.0); glVertex3f(-2.0,2.0,0.0);
    glEnd();
}
```

```

glBindTexture(GL_TEXTURE_2D,texture[1]);
glBegin(GL_QUADS);
    // Superior
    glTexCoord2f(0.0,0.0);glVertex3f(-2.5,2.5,-.5);
    glTexCoord2f(1.0,0.0);glVertex3f(2.5,2.5,-.5);
    glTexCoord2f(1.0,1.0);glVertex3f(2.0,2.,-.5);
    glTexCoord2f(0.0,1.0);glVertex3f(-2,2.,-.5);
    // a mi izquierda
    glTexCoord2f(0.0,0.0);glVertex3f(-2.5,2.5,-.5);
    glTexCoord2f(1.0,0.0);glVertex3f(-2.0,2.,-.5);
    glTexCoord2f(1.0,1.0);glVertex3f(-2.0,-2.,-.5);
    glTexCoord2f(0.0,1.0);glVertex3f(-2.5,-2.5,-.5);
    // Inferior
    glTexCoord2f(0.0,0.0); glVertex3f(-2.5,-2.5,-0.5);
    glTexCoord2f(1.0,0.0); glVertex3f(2.5,-2.5,-0.5);
    glTexCoord2f(1.0,1.0); glVertex3f(2.0,-2,-0.5);
    glTexCoord2f(0.0,1.0); glVertex3f(-2.0,-2.,-0.5);
    //a mi derecha
    glTexCoord2f(0.0,0.0); glVertex3f(2.5,2.5,-.5);
    glTexCoord2f(1.0,0.0); glVertex3f(2.0,2.,-.5);
    glTexCoord2f(1.0,1.0); glVertex3f(2.0,-2.,-.5);
    glTexCoord2f(0.0,1.0); glVertex3f(2.5,-2.5,-.5);
    //cuadritos
    glTexCoord2f(0.0,0.0); glVertex3f(-.05,-2.0,0.5);
    glTexCoord2f(1.0,0.0); glVertex3f(.05,-2.0,0.5);
    glTexCoord2f(1.0,1.0); glVertex3f(.05,2.0,0.5);
    glTexCoord2f(0.0,1.0); glVertex3f(-.05,2.0,0.5);
    glTexCoord2f(0.0,0.0); glVertex3f(-2.0,-.05,0.5);
    glTexCoord2f(1.0,0.0); glVertex3f(2.0,-.05,0.5);
    glTexCoord2f(1.0,1.0); glVertex3f(2.0,.05,0.5);
    glTexCoord2f(0.0,1.0); glVertex3f(-2.0,.05,0.5);
glEnd();
glFlush();
}

void reshape(int width,int height) {
    glViewport(0,0,width,height);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-4,4,-4,4,-4,4);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}

void keyboard(unsigned char key,int x,int y)
{
    if (key==27) exit(0);
}

int main(int argc,char **argv) {
    glutInit(&argc,argv);
    glutInitWindowSize(412,412);
    glutInitDisplayMode(GLUT_RGB|GLUT_SINGLE);
    glutCreateWindow("Ventana textura sintética");
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutKeyboardFunc(keyboard);
    glutMainLoop();
    return 0;
}

```

